Keeping Secrets In Hardware

The Microsoft Xbox™ Case Study

S-BOXes and Xboxes

Andrew "bunnie" Huang, MIT
bunnie@alum.mit.edu

Outline

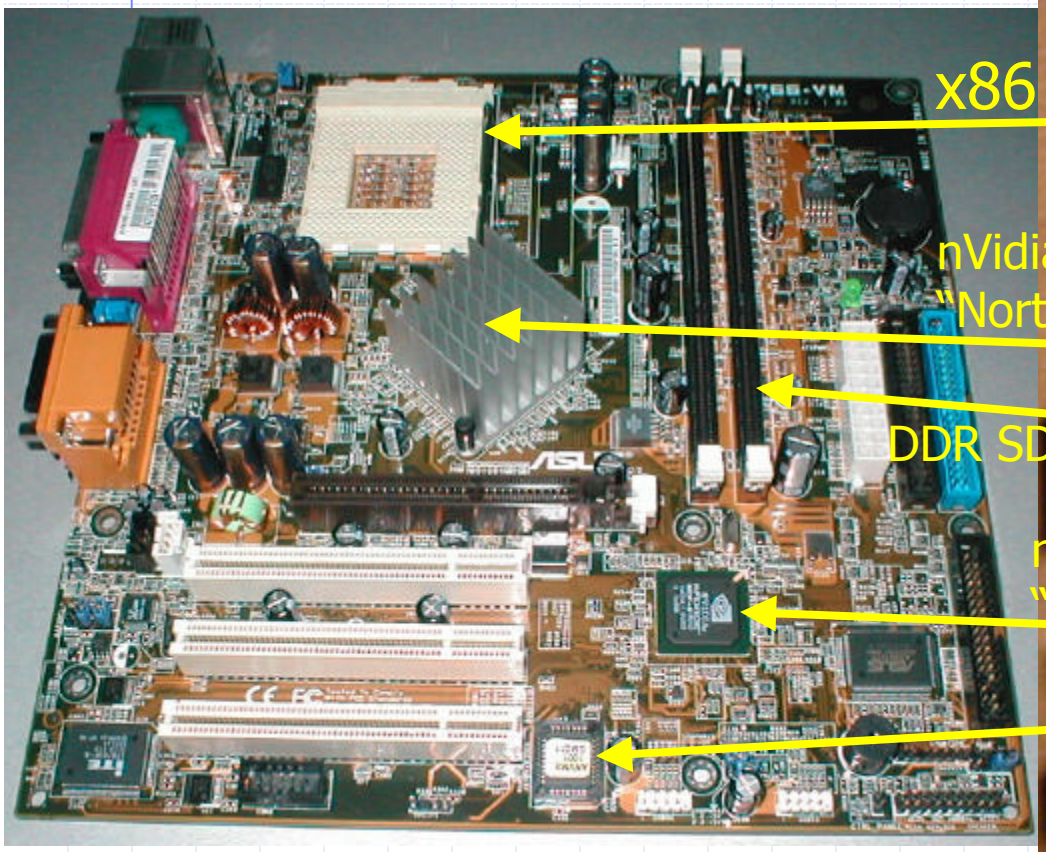
- Background
  - Subject hardware
  - Security motive
- Xbox™ Security overview
- Reverse engineering
  - Focus on process and methodology
- Lessons learned
  - Summary of known flaws
  - Summary of possible countermeasures

What is an Xbox?

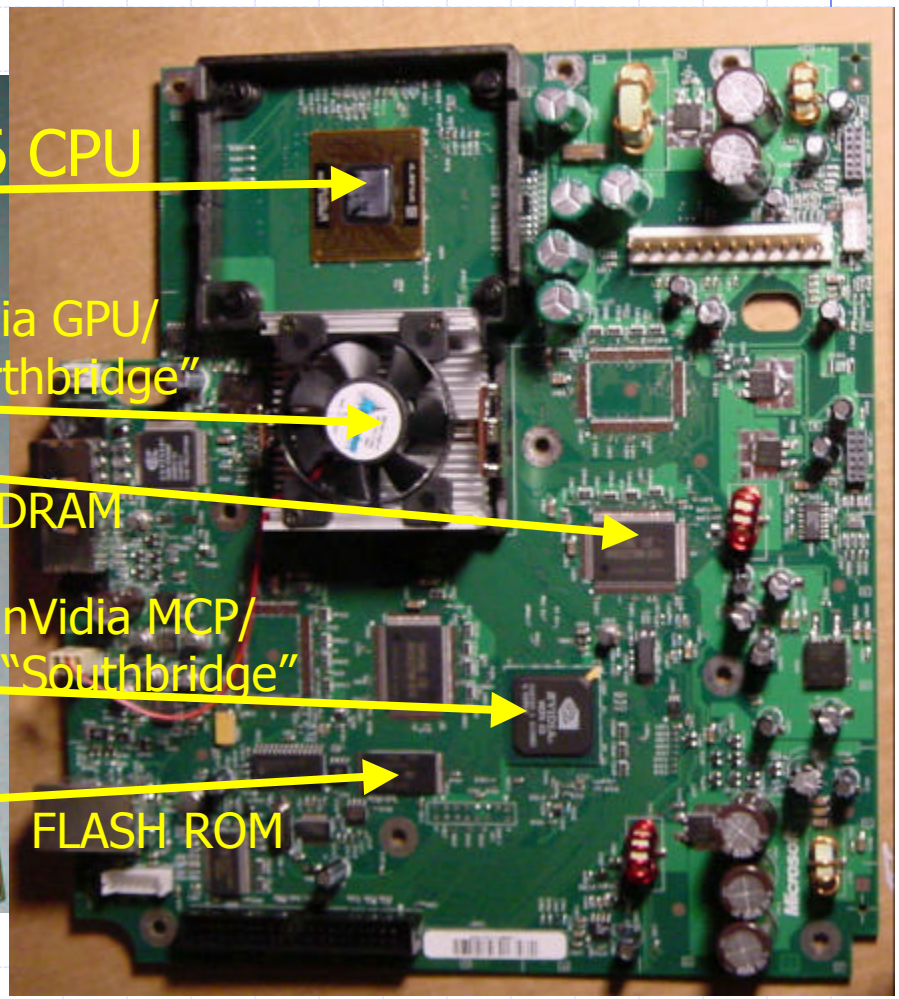
- Xbox is an embedded PC
  - 733 MHz Intel Pentium III-class processor
  - nVidia nForce-derivative chipset
  - 64 MB DDR SDRAM
  - 100 Base-T ethernet port
  - VGA graphics capability
  - USB ports
  - 10 GB IDE hard drive
  - IDE DVD-ROM drive

Comparison to Stock PC Hardware

ASUS A7N266-VM



Xbox Motherboard



Picture from
http://www.ocmodshop.com/asusnforce/topboard.jpg

Security Rationale: Economics

- Hardware is sold at a loss
  - "Loss Leader"
  - Make up the difference in sales of games, services

Economic Details

- Sell about 20 games to break even
  - US$100-200 lost per Xbox console
  - Microsoft makes ~$7/title for third-party games
  - Microsoft makes about 3-4x more on first-party titles
- Assuming 1:2 first party:third party sale ratio
  - Over US$1000 in software

How Much Security?

- Sufficient deterrent to ensure that:
  - $1000 in games, or $200 in game services are purchased over console lifetime
  - On-line gaming experience is enjoyable
    - A billion-dollar investment on Microsoft's part

Security Rationale: Summary

- Prevent the following key scenarios:
  - Game copying
  - Game cheating
    - Ensure an enjoyable on-line gaming experience
  - Emulation
    - Stock PC booting a copied Xbox game
    - Modified PC booting a copied Xbox game
  - Conversion to stock PC
    - Subsidized Windows platform
    - Linux/freeware platform
    - Embedded controller

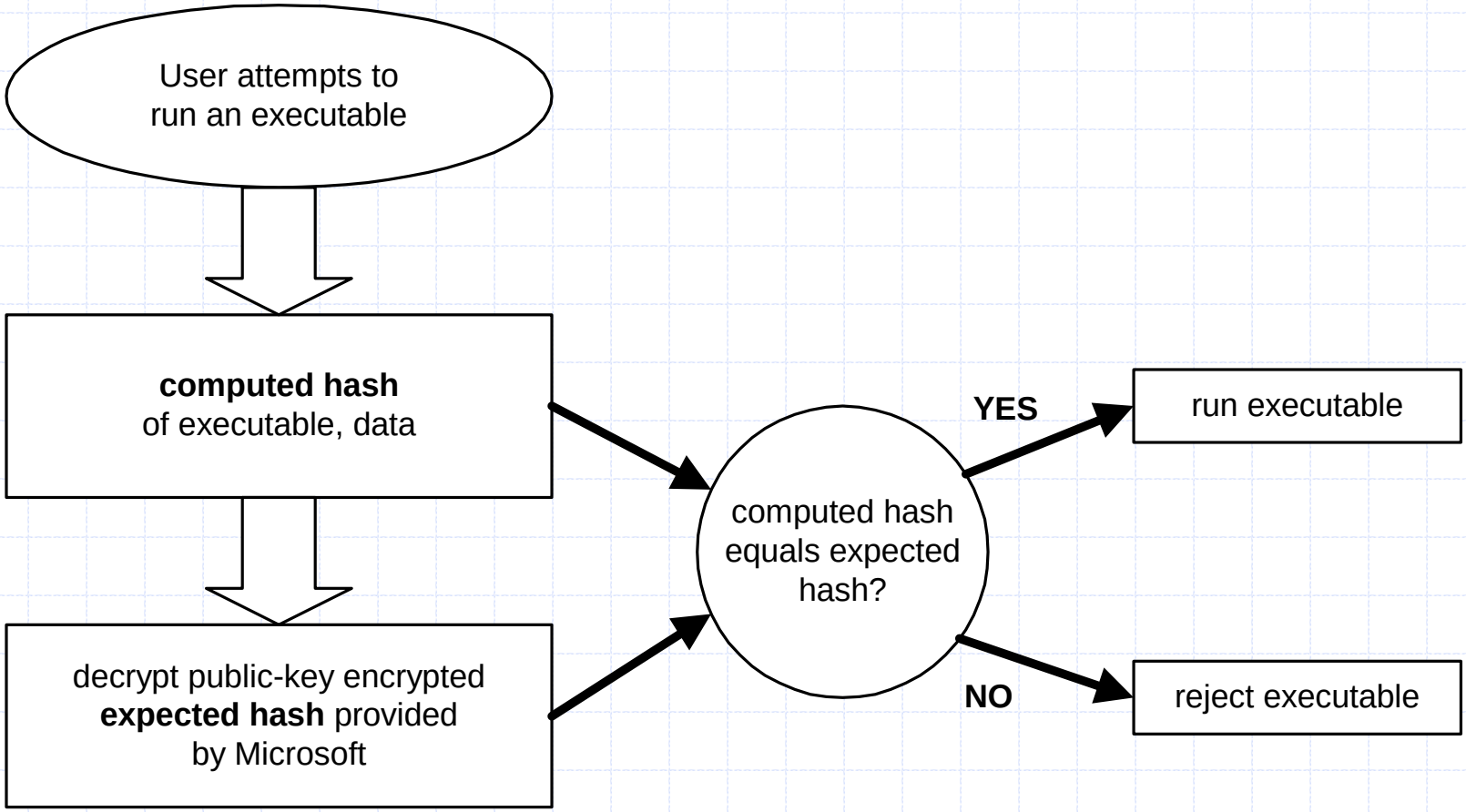
Xbox Security Overview

- Xbox is a Trusted PC Platform
  - Comparable *in spirit* to Palladium™, TCPA
  - Hardware is trusted, all executables digitally signed and verified prior to execution
- Physical copy protection
  - 2-Layer DVD- format block scrambling
  - 2-Layer DVDs are difficult to copy
- Encrypted network connections
  - No details available yet, Xbox Live not yet launched
- Minimal perimeter security, tamper evidence

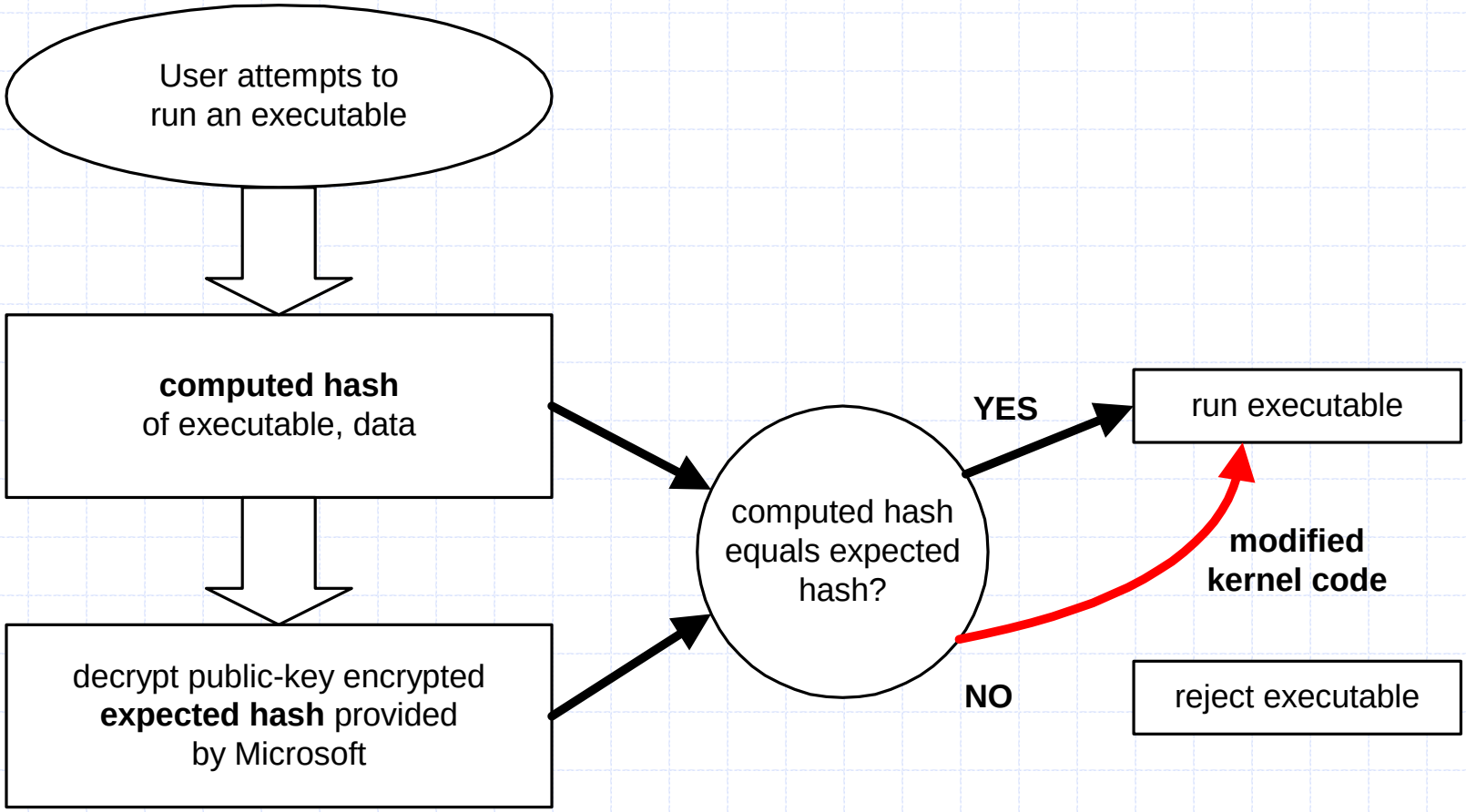
Focus on Trust Mechanism

- Trustable hardware is a cornerstone of Xbox security
  - If hardware is compromised, there is no security

Why Trust is Re uired



Why Trust is Re uired



Establishing Trust

- Re uirements
  - The program counter PC is always within a trusted code region, starting with the reset vector
  - All code and data is verified against signed hashes before being accepted
  - Code and hardware is free of bugs
    - i.e., buffer and segment overruns, protocol weaknesses
  - Hardware is inviolable
    - Intrusion detection at a minimum
    - Tamper resistance preferable

Establishing Trust

Microsoft does these

- Re uirements
  - The program counter PC is always within a trusted code region, starting with the reset vector
  - All code and data is verified against signed hashes before being accepted
  - Code and hardware is free of bugs
    - i.e., buffer and segment overruns, protocol weaknesses
  - Hardware is inviolable
    - Intrusion detection at a minimum
    - Tamper resistance preferable

Establishing Trust

- Re uirements
  - The program counter PC is always within a trusted code region, starting with the reset vector
  - All code and data is verified against signed hashes before being accepted
  - Code and hardware is free of bugs
    - i.e., buffer and segment overruns, protocol weaknesses
  - Hardware is inviolable
    - Intrusion detection at a minimum
    - Tamper resistance preferable

Attempts to do these

Establishing Trust

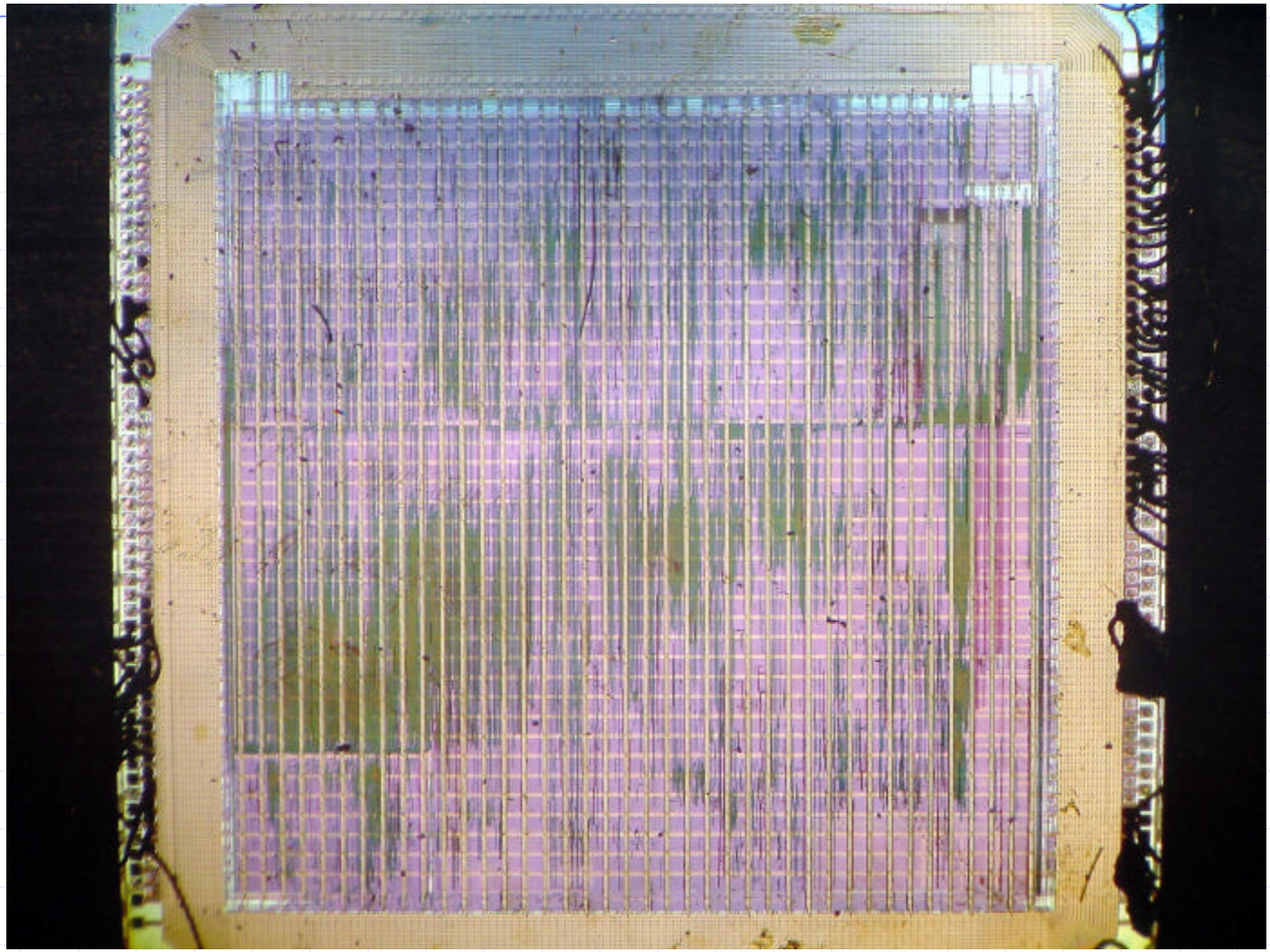
- Re uirements
  - The program counter PC is always within a trusted code region, starting with the reset vector
  - All code and data is verified against signed hashes before being accepted
  - Code is free of bugs
    - i.e., buffer overruns, protocol weaknesses
  - Hardware is inviolable
    - Intrusion detection at a minimum
    - Tamper resistance preferable

Fails to do this

Root of Trust

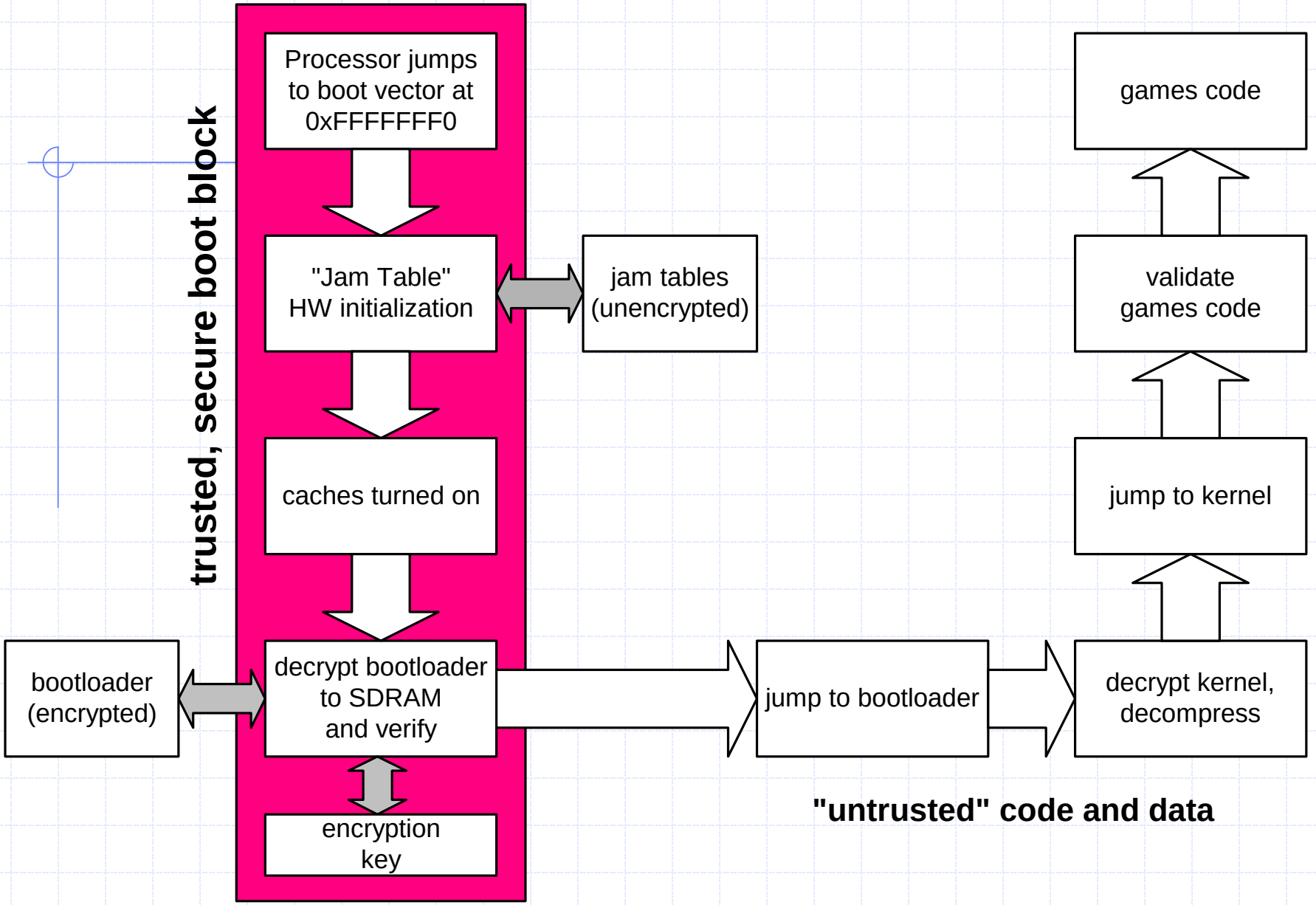
- Linear trust mechanism
  - Chain of trustable, verified code, starting with the secure boot block
- Secure boot block details
  - Reset vector/init code is contained in a tamper-resistant module
    - ROM overlay within the system peripherals ASIC "southbridge" ASIC
    - Southbridge ASIC implemented in 0.15?, 6 or 7 layers of metal
      - Very hard to probe or modify

Tamper Resistance?



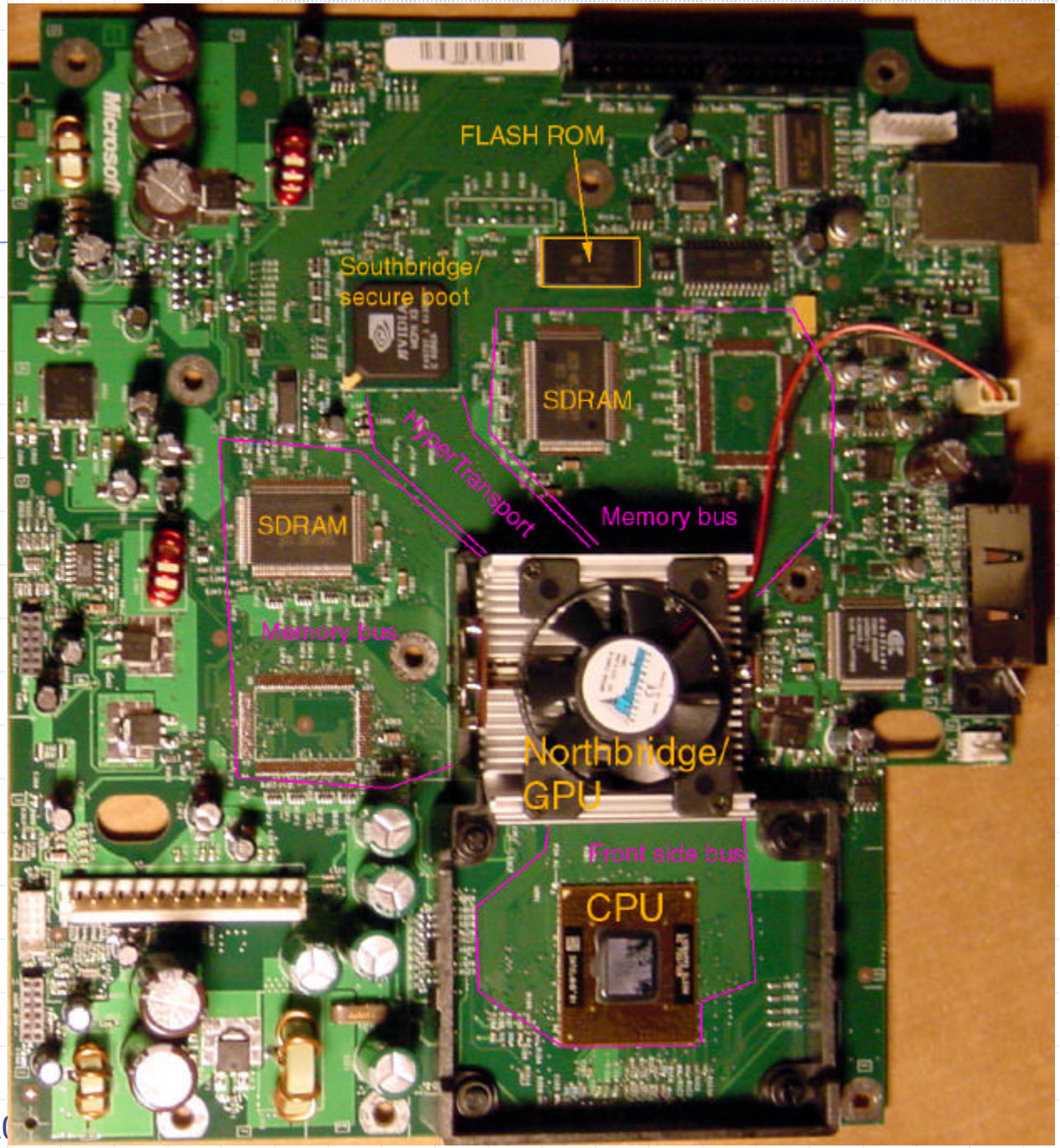
Transferring the Trust

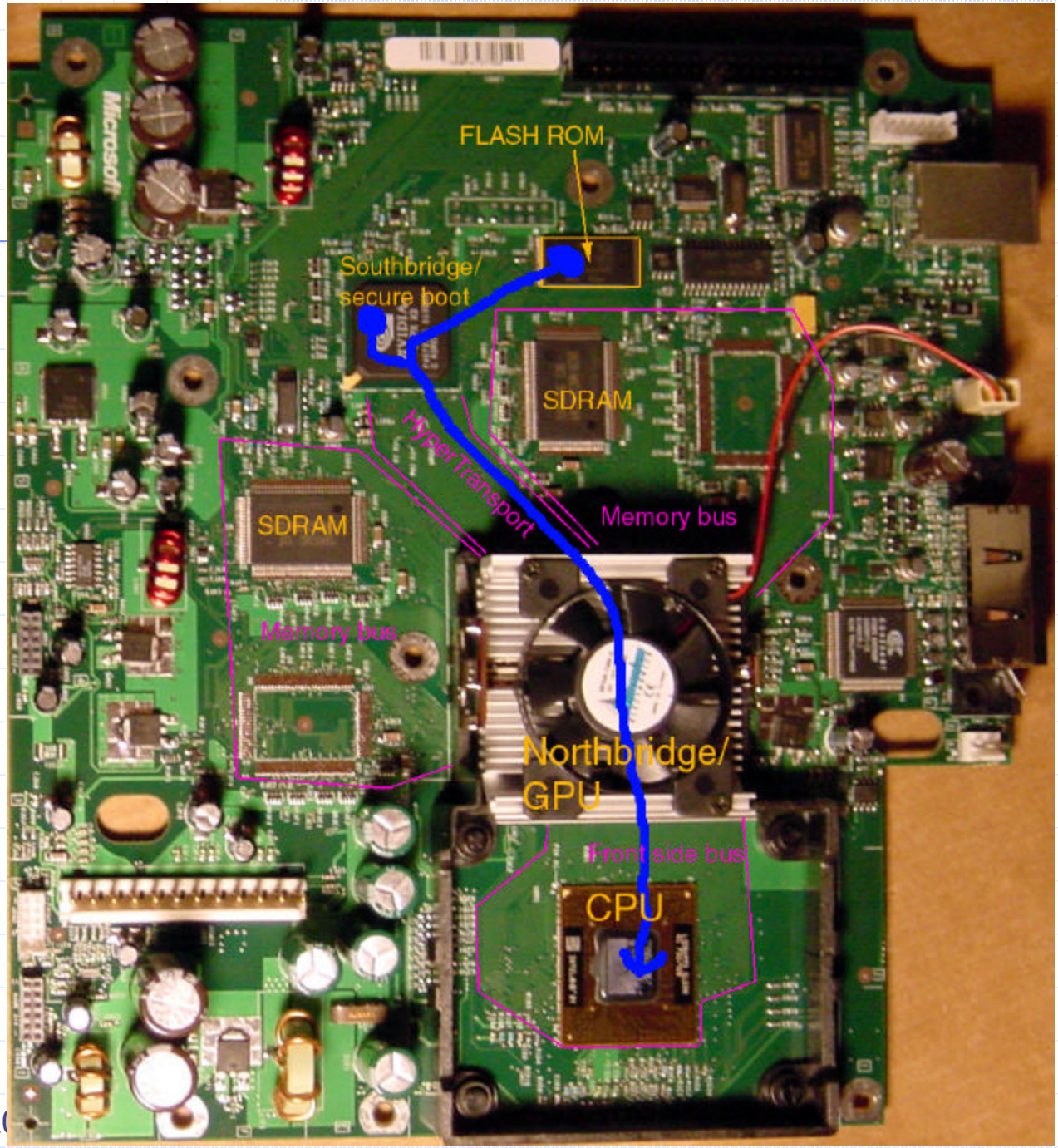
- RC4/128 used to encrypt bootloader image
  - RC4/128 is a stream cipher
    - A ciphertext modification will corrupt the remainder of the plaintext stream
  - Simple "magic number" at the end of the bootloader image, checked to verify integrity
- So long as the RC4/128 key is secret, attackers are unlikely to generate a valid false bootloader image
  - Secondary bootloader continues to transfer trust through verification of digitally signed binaries


trusted, secure boot block
Processor jumps to boot vector at 0xFFFFFFF0
"Jam Table" HW initialization
jam tables (unencrypted)
caches turned on
bootloader (encrypted)
decrypt bootloader to SDRAM and verify
encryption key
jump to bootloader
decrypt kernel, decompress
jump to kernel
validate games code
games code
"untrusted" code and data

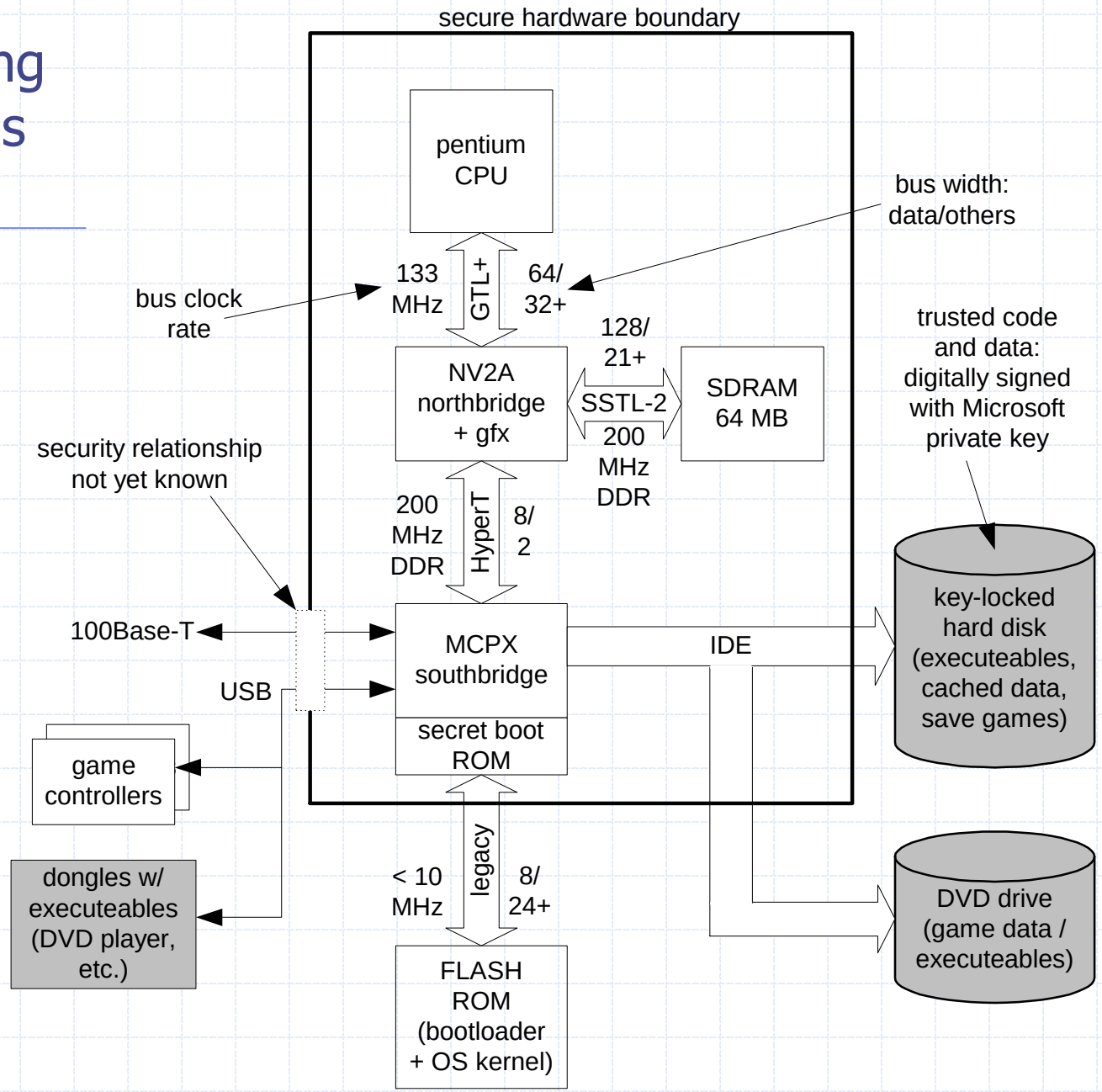
Breaking the Trust

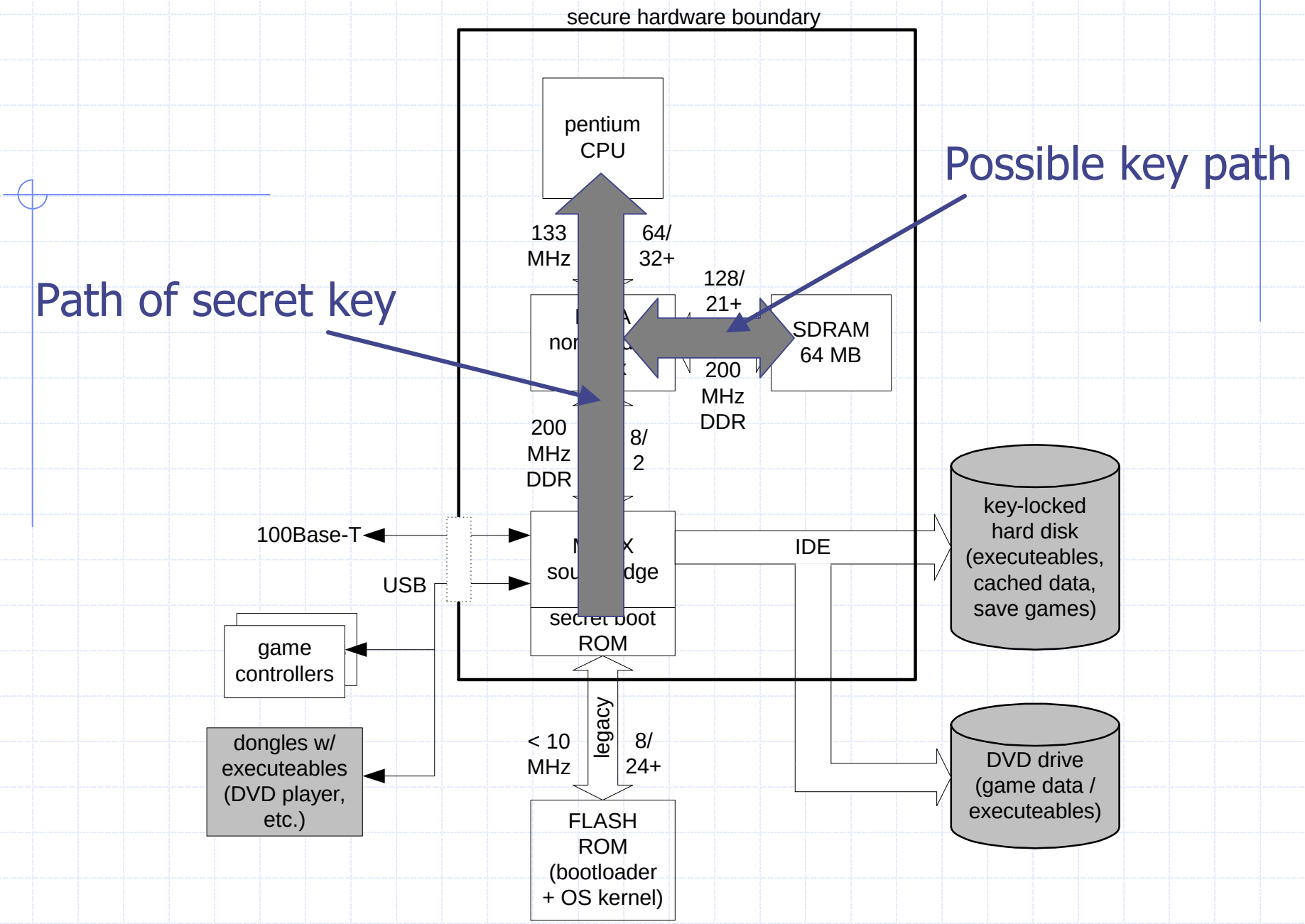
- Discovery of secret key breaks the trust
  - Secure boot block was discovered when certain ROM mods did not affect operation
  - Key was extracted by sniffing internal busses


FLASH ROM
Southbridge/
secure boot
SDRAM
HyperTransport
Memory bus
SDRAM
Memory bus
Northbridge/
GPU
Front side bus
CPU

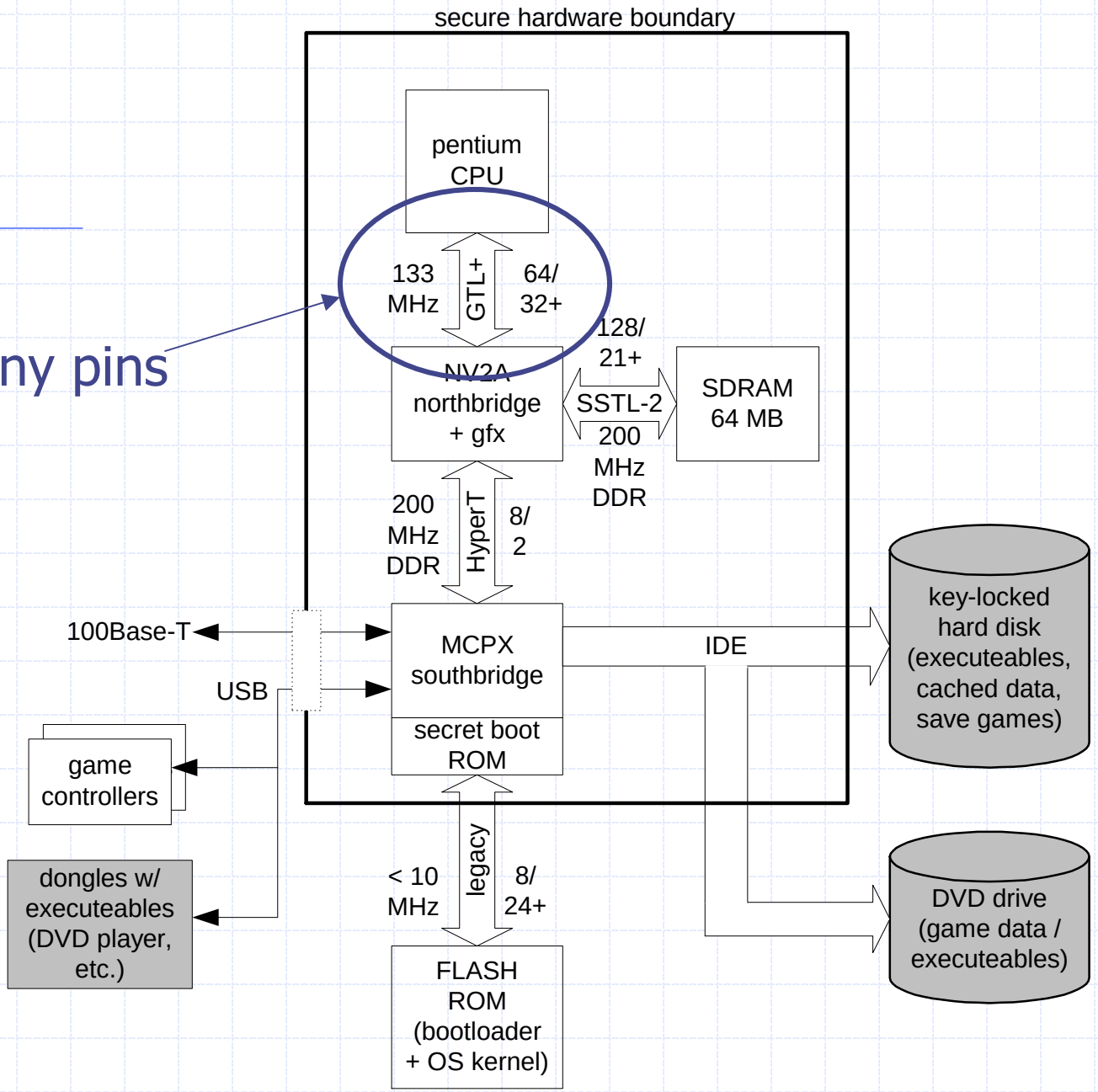

FLASH ROM
Southbridge/
secure boot
SDRAM
HyperTransport
Memory bus
SDRAM
Memory bus
Northbridge/
GPU
Front side bus
CPU

Bus Sniffing Candidates

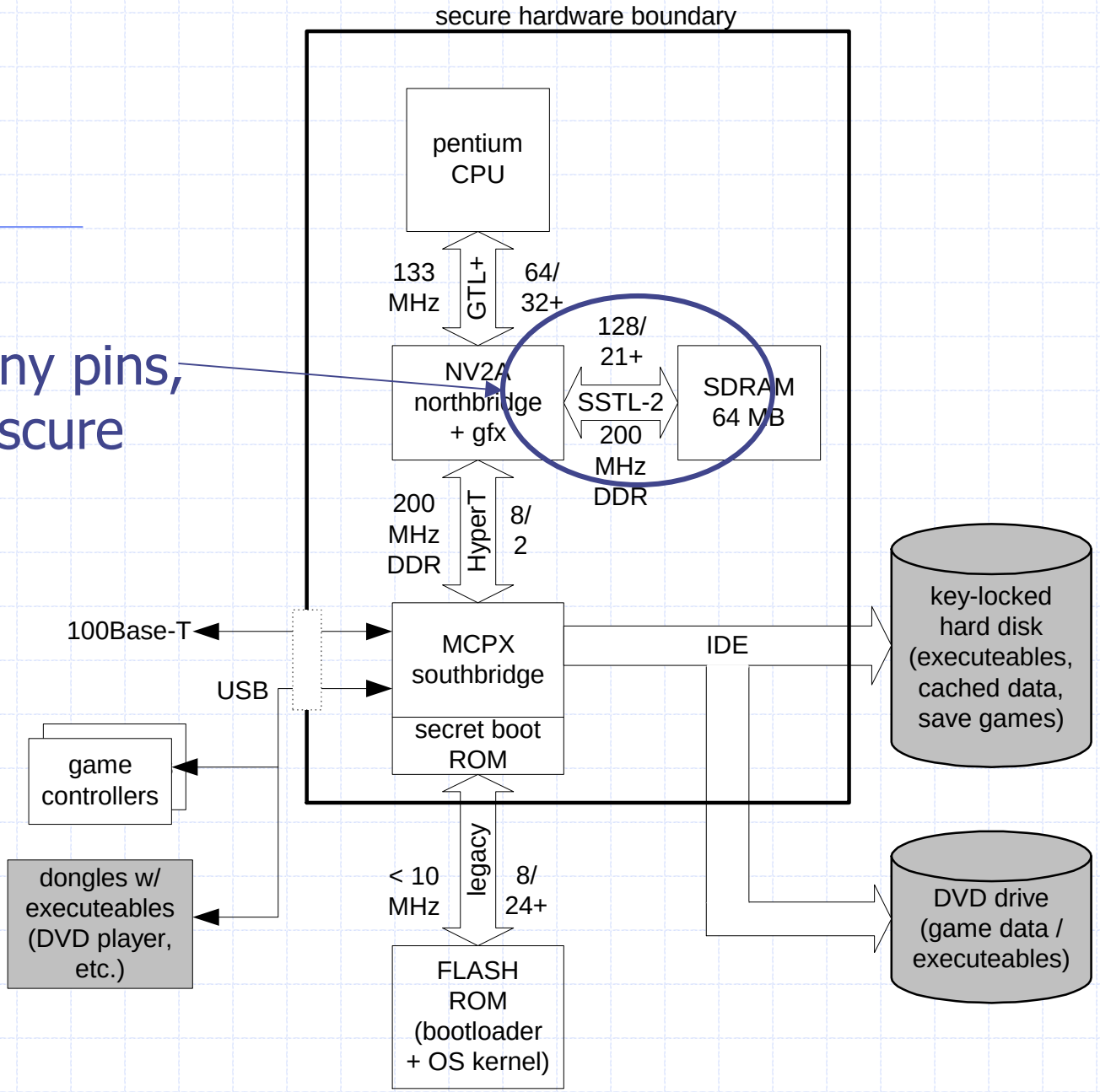



secure hardware boundary
pentium CPU
133 MHz
64/ 32+
128/ 21+
SDRAM 64 MB
200 MHz DDR
200 MHz DDR
8/ 2
100Base-T
USB
IDE
key-locked hard disk (executeables, cached data, save games)
game controllers
dongles w/ executeables (DVD player, etc.)
secret boot ROM
< 10 MHz
legacy
8/ 24+
DVD drive (game data / executeables)
FLASH ROM (bootloader + OS kernel)
Path of secret key
Possible key path

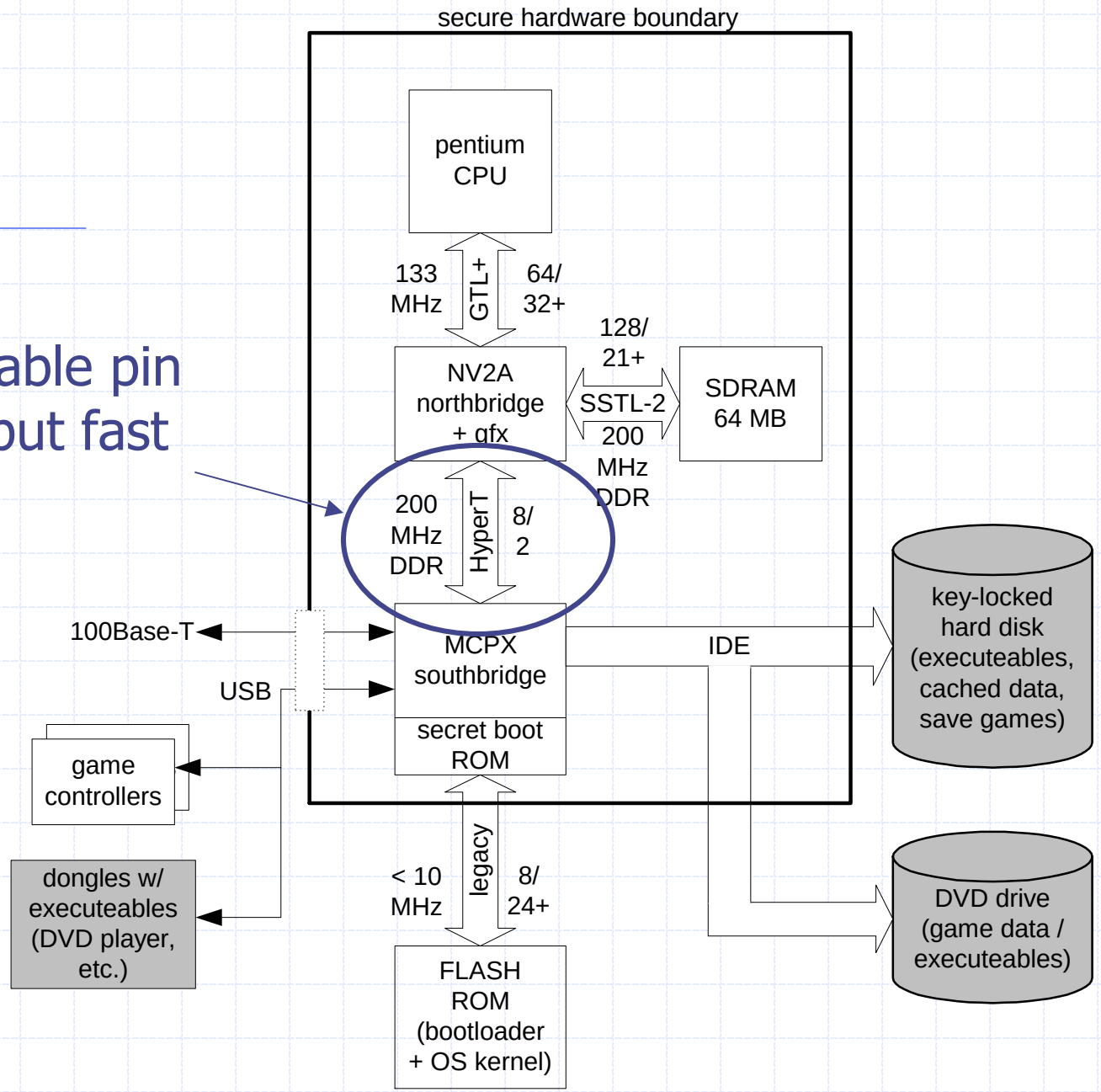
Too many pins



Too many pins, fast, obscure layout

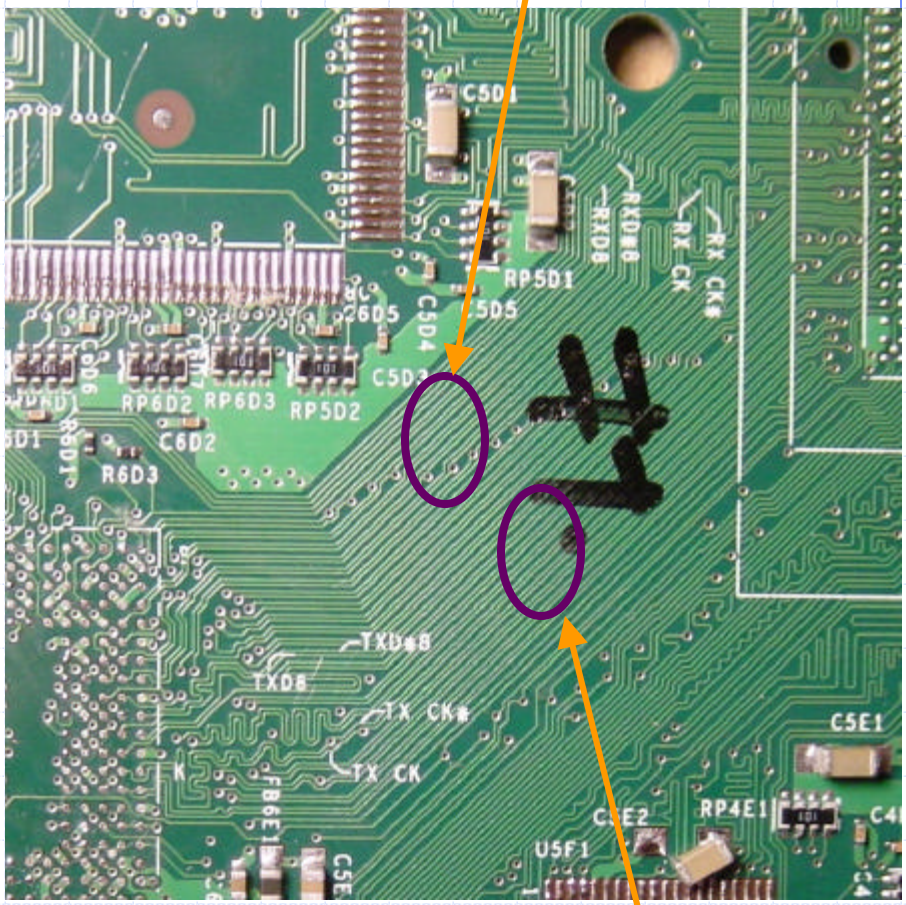


Reasonable pin count, but fast

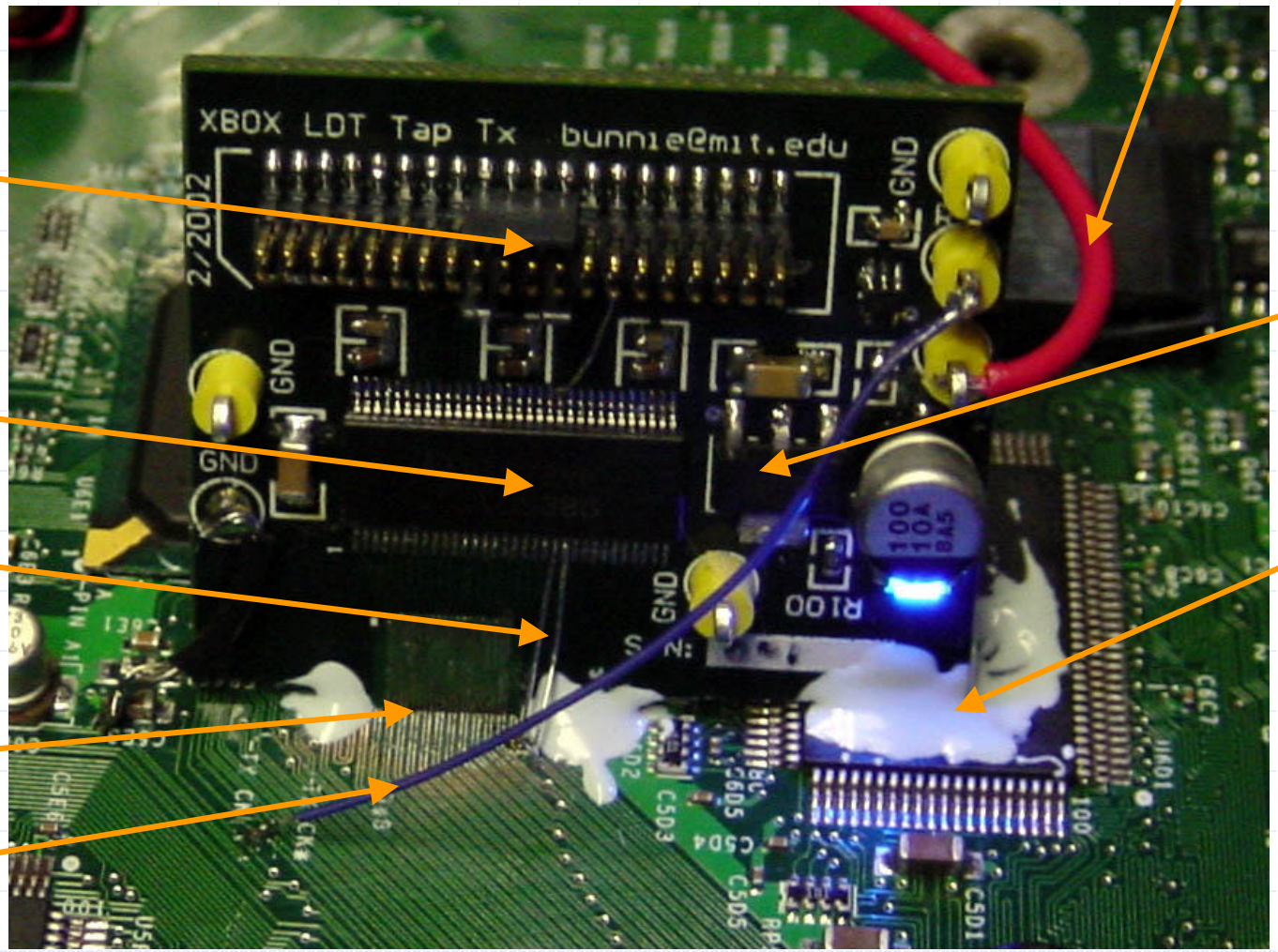


HyperTransport Bus

- Favorable board layout, pin count
  - Fabricate pitch-matched tap board
- High speed
  - Use high-end FPGA or logic analyzer



Custom Tap Board



CTT to FPGA

LVDS-TTL converter

Last-minute signal pair

Pitch-matched HT connector

Reset signal

5V power in

3.3V Local regulator

Epoxy in place

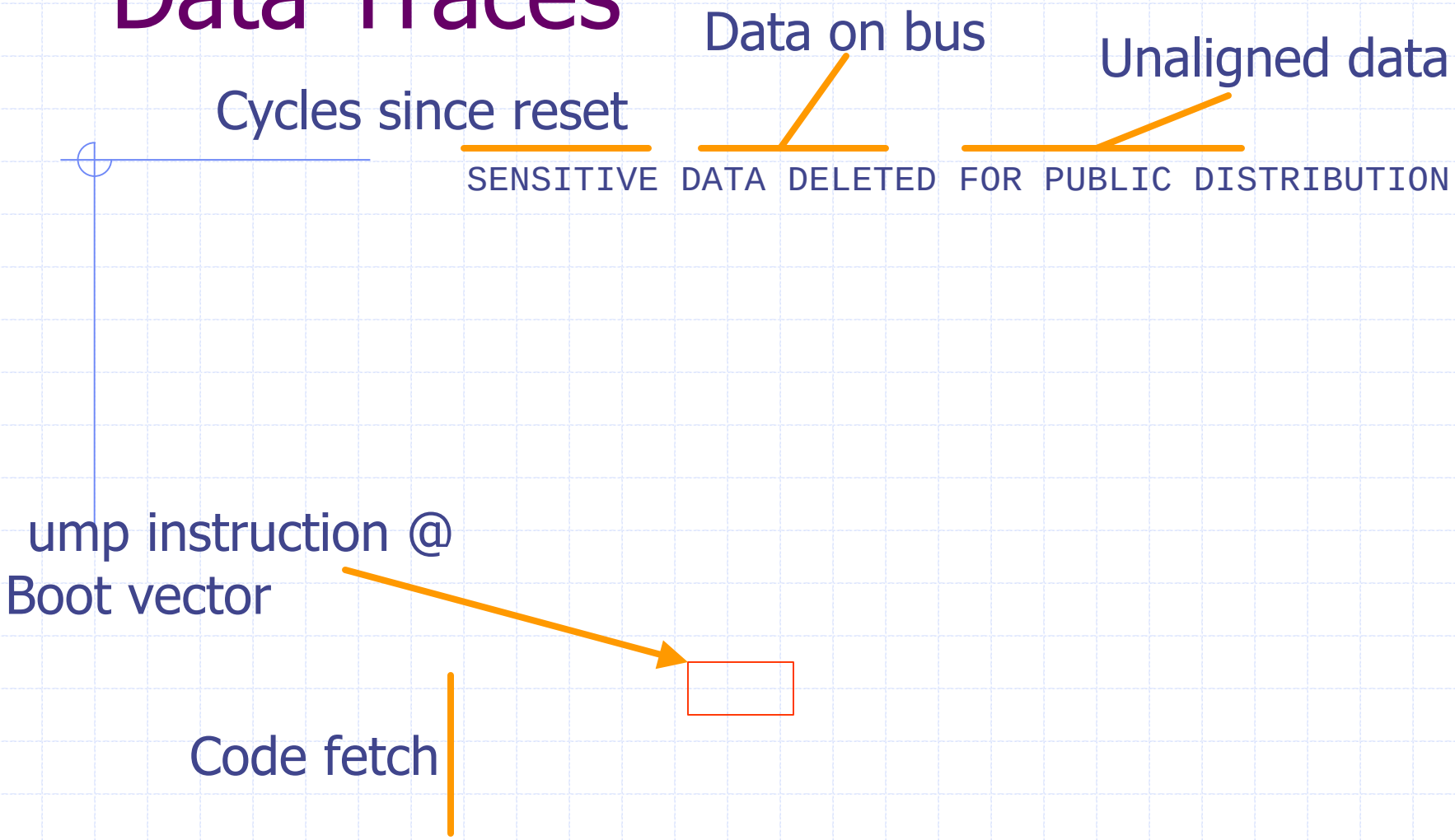
Tap Board

- Board adapts HyperTransport bus to existing hardware
  - Virtex-E FPGA board developed for my thesis
- Clean-sheet tap board would look different
  - Virtex-II FPGA directly on tap board
  - Would cost $50-$100 to fabricate

Analyzing the Bus

- Traces of data collected, synchronized to power-on reset
- Ciphertext sorted from code by histogramming and eyeballing
- Data in traces organized by cache line
  - Code path was patched together using a disassembler and cache line groupings

Data Traces


Data on bus
Unaligned data
Cycles since reset
SENSITIVE DATA DELETED FOR PUBLIC DISTRIBUTION
ump instruction @
Boot vector
Code fetch

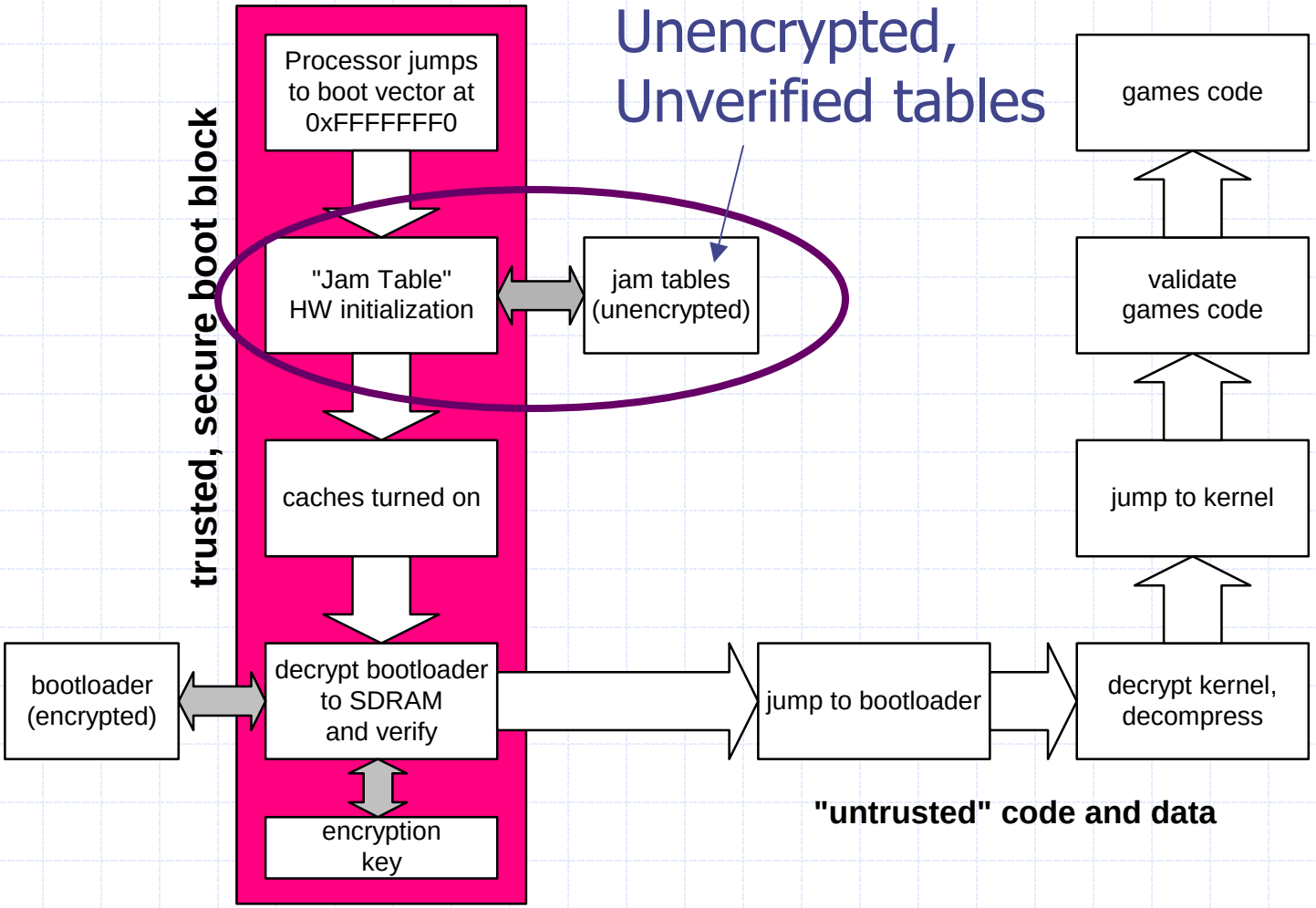
Piecing it Together

- Traces assembled into an image of the secure boot ROM
  - Secure boot ROM image contains
    - RC4/128 key
    - Magic number check

Fragile Trust

- All Xboxes use the same secret key
  - One key extraction applies to all boxes
  - Debug and test features on the Xbox motherboard enable easy ROM override
    - Easy to create, encrypt, and deploy mass uantities of untrusted hardware

Backdoors Galore



amtable Interpreter

- What it is
  - Bytecode interpreter
  - Orchestrates dependencies and decisions re uired for machine initialization
- What it can do
  - Reads and writes to PCI, memory, I/O space
  - Conditional jumps, indirect addressing

amtable Attacks

- amtables are unencrypted and unverified
  - Can perform attacks without crypto
  - Two-phase soft-reset attacks to read out plaintext
    - Allow machine to power up normally once, then soft reset with a new jam table that copies code to an insecure location courtesy visor

amtable Attacks II

- amtable weakness hardware bugs allows program counter to be seized
  - Secure boot block jumps to 0xFFFF FFFA when a bad ciphertext image is encountered
  - PC will roll over from 0xFFFF FFFF to 0x0000 0000 without an exception
  - 0x0000 0000 is in SDRAM memory
  - Use jamtable to write at 0x0000 0000 a jump instruction to an insecure FLASH region, and corrupt ciphertext image to sieze the PC
  - Courtesy visor

Lessons Learned

- Avoid symmetric ciphers in this scenario
  - Difficult to guarantee secrecy of key
  - Cost of ASIC mask sets, lead time make key rotation expensive and difficult
- Use hashes to verify **all** code and data regions
- Complex protocols such as x86/PC initialization are difficult to secure

Alternative Solution

- Use digital signatures to verify the FLASH ROM contents
  - Store signature in off-chip EEPROM
  - Users cannot run false code without signer's private key
  - Does not prevent plaintext snooping
  - Can be defeated with a bus override attack
    - A set of precisely timed pulses on the HyperTransport bus can alter the reset vector

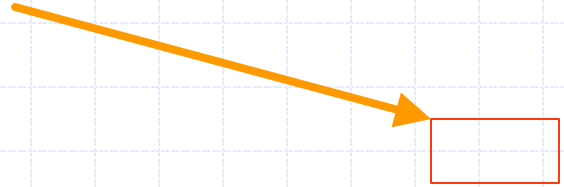
Bus Override Attack

Cycles since reset

Data on bus

SENSITIVE DATA DELETED FOR PUBLIC DISTRIBUTION

ump instruction @
Boot vector



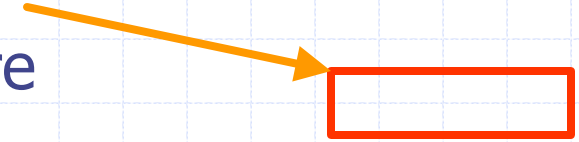
Bus Override Attack

Cycles since reset

Data on bus

```
SENSITIVE DATA SCRUBBED
```

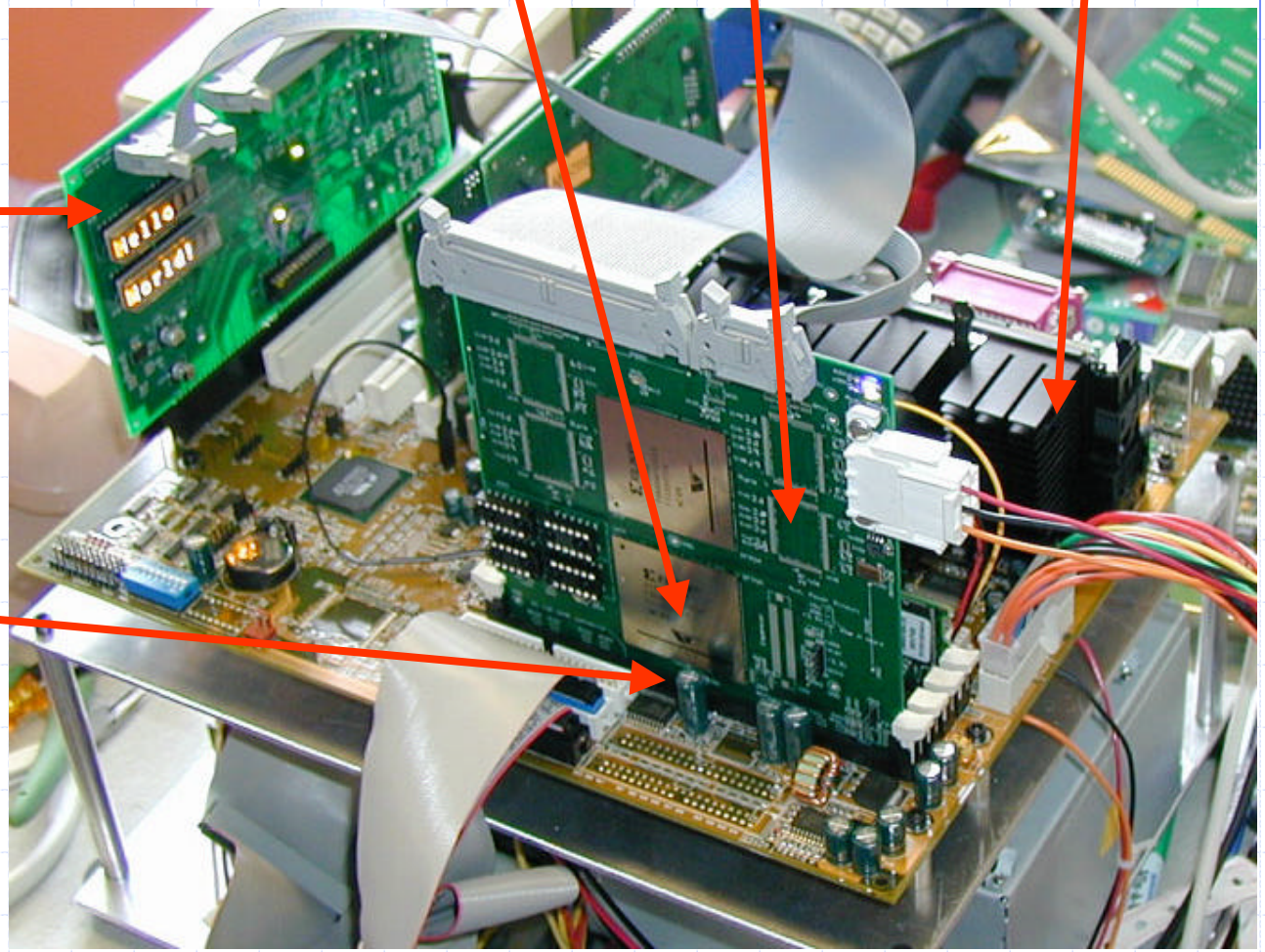
Override cycle 22526 with jump opcode to insecure code space



Alternative Solution, Cont'd

- Use digital signatures to verify the FLASH ROM contents
  - Can be defeated with a snoop modify memory
    - Most effective in a PC using standard memory sockets
    - Present trust introspection routines with benign code images
    - Present malicious memory image at other times
    - Also use to snoop and extract plaintexts
    - Snoop-RAM can be fairly inexpensive to manufacture
    - Inspired by entries about Palladium in Seth Schoen's online diary

"Snoop-RAM"


Memory
Interceptor FPGA
Stock PC motherboard
Snoop capture card
Standard SDRAM DIMM socket

Even More Security Measures

- Embed secret boot block on processor silicon
  - Bus override attack extremely difficult
  - Possible Vcc, photonic attacks c.f. R. Anderson / smartcards
- Employ tamper-evidence
  - Expect tampering, disable system if tampered
  - Possible yield hit and field service issues
- Physical tamper-resistance
  - Potting, tamper-detecting membranes
  - Expensive, impractical, thermal issues

Other Ideas

- Encrypt all chip-chip busses
  - Severe power consumption implications
  - Reliability can be impacted
  - Performance is degraded

Other Ideas

- Don’t use a PC
  - “Security through obscurity” c.f. Nintendo
  - Patent proprietary formats
    - Well-understood legal protections
  - The end goal is not crypto-security
    - An economic or legal barrier is a sufficient deterrent
    - Unfortunately, the DMCA presents a significant psychological threat to many researchers in the US

Summary

- Xbox is a PC architecture with trust enhancements
  - Trust relies on the secrecy of a key, contained in the user hardware
    - Demonstrated key extraction
  - Other protocol attacks i.e., jamtable attacks can bypass the trust mechanism
- Creating a trusted PC architecture is not trivial
  - Like turning a college campus into a fortress

Thank you for your attention.