

FREUD Applications of FIB

Invasive FIB Attacks and Countermeasures in Hardware Security Devices

Valery Ray

vray@partbeamsystech.com

FIB User Group, Washington DC, USA



F R E U D

**Functional Reverse Engineering of
Undocumented Devices[©]**

**Extraction of functional codes,
algorithms, data, and keys from
secured hardware**

Outline

- ◆ **Targeted Devices and Applications**
- ◆ **Workflow of FIB “invasion”, challenges and tricks**
- ◆ **Signal extraction and injection**
- ◆ **Limitations of FIB instrumentation**
- ◆ **Countermeasures against FIB methods**

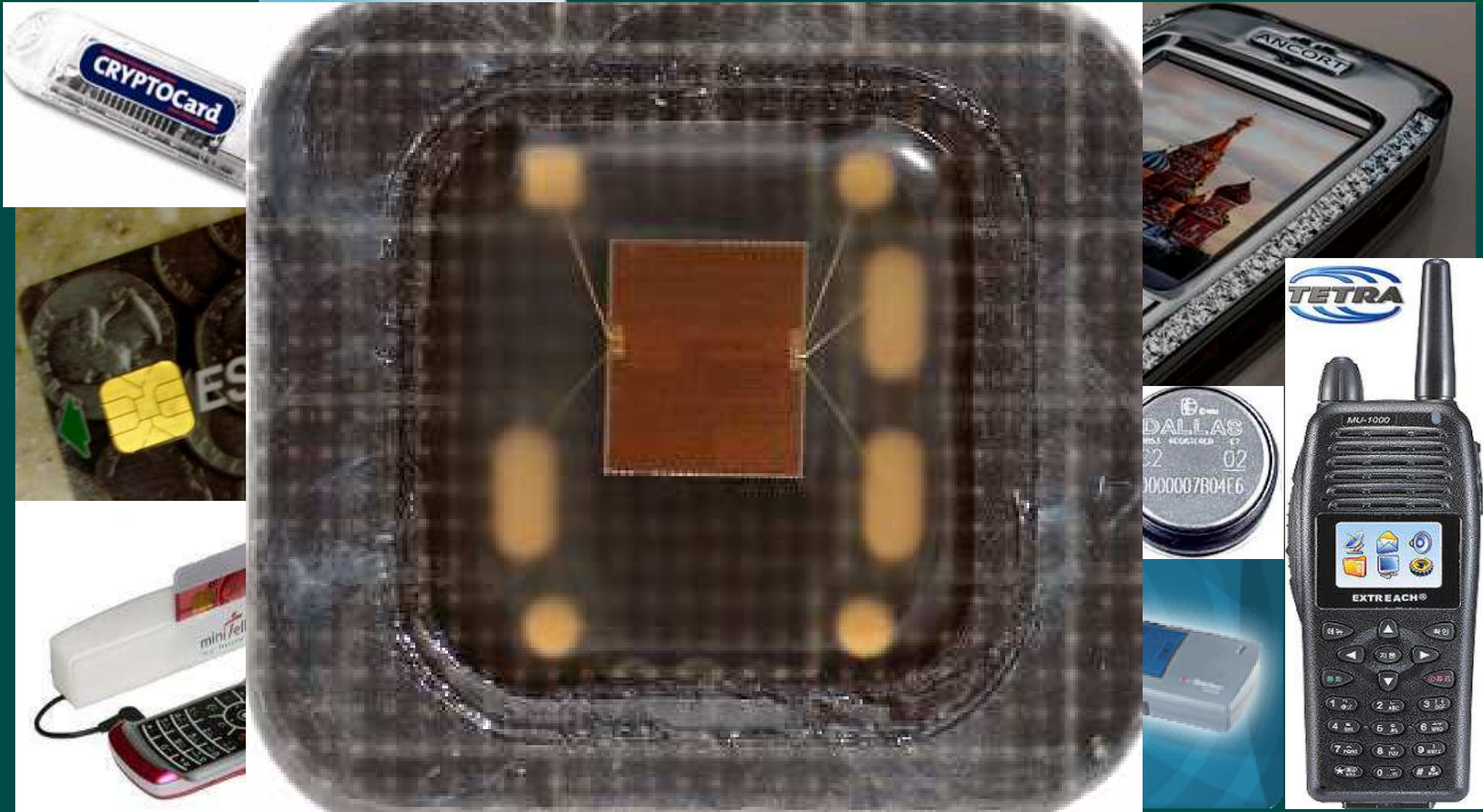
Typical Targeted Devices: μ C in Distributed Security and Encryption Applications



3/8/2009

FIB User Group, Washington DC, USA

Typical Targeted Devices: μ C in Distributed Security and Encryption Applications



3/8/2009

FIB User Group, Washington DC, USA

5

Workflow of FIB “Invasion”

- ◆ **Navigation on undocumented secure devices**
- ◆ **Capturing layout and localizing nodes**
- ◆ **Bypassing protective shields**
- ◆ **Making contacts, extracting data, injecting signals**

Navigation on Secure Devices

- Shields prevent direct navigation with optics
- Have to use sacrificial device to localize nodes
- Two steps of localization – coarse and precise
- Dual positioning: coordinates and local reference

Shield images

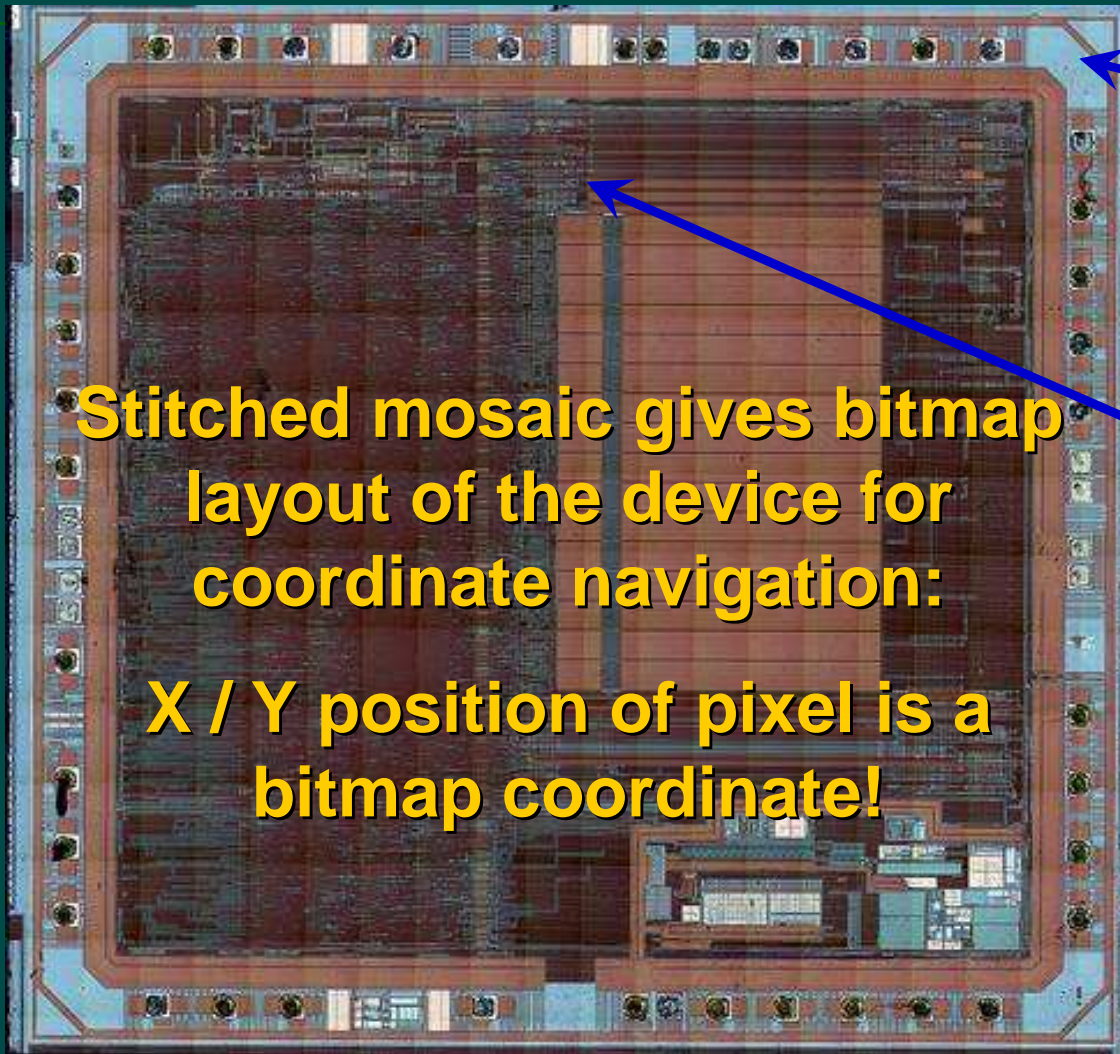
© Christopher Tarnovsky

www.flylogic.net

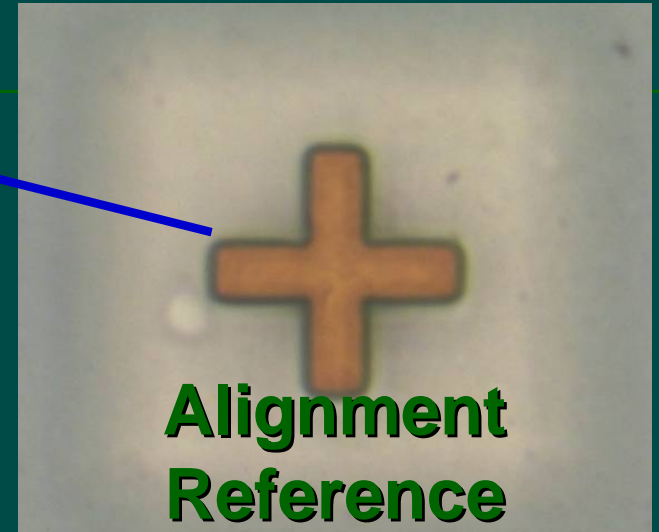
Coarse Localization on Sacrificial Device (s)

- ◆ **Remove shield (wet chemistry or RIE)**
- ◆ **Scan device under Optical Microscope, stitch mosaic bitmap, locate nodes, define coordinates**
- ◆ **Establish references for coordinate conversion**
- ◆ **Convert bitmap coordinates to FIB stage position**

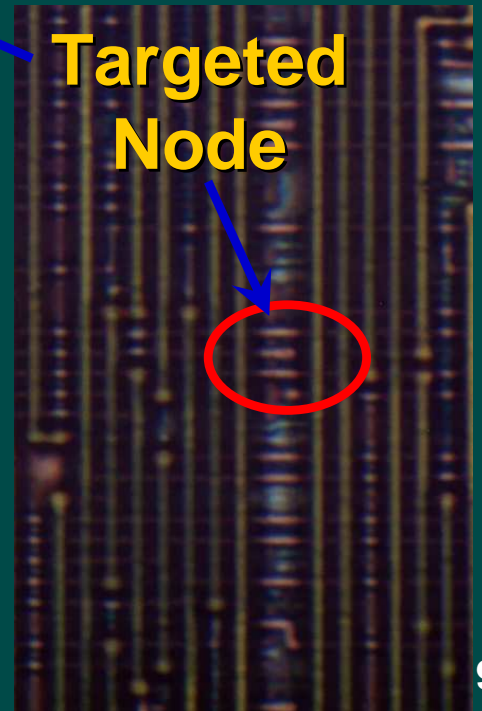
Layout Capture and Node Localization



**Stitched mosaic gives bitmap layout of the device for coordinate navigation:
X / Y position of pixel is a bitmap coordinate!**

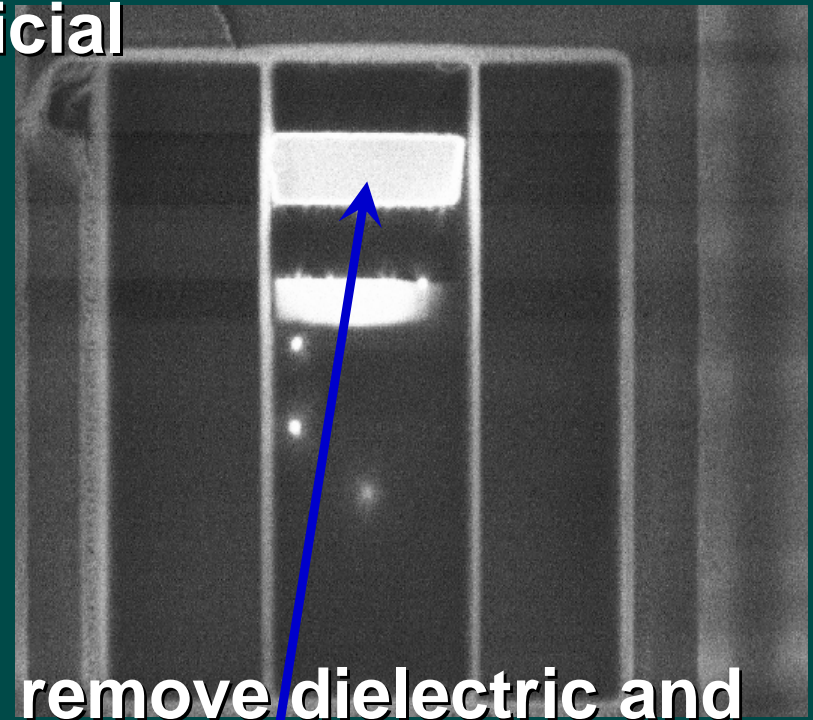
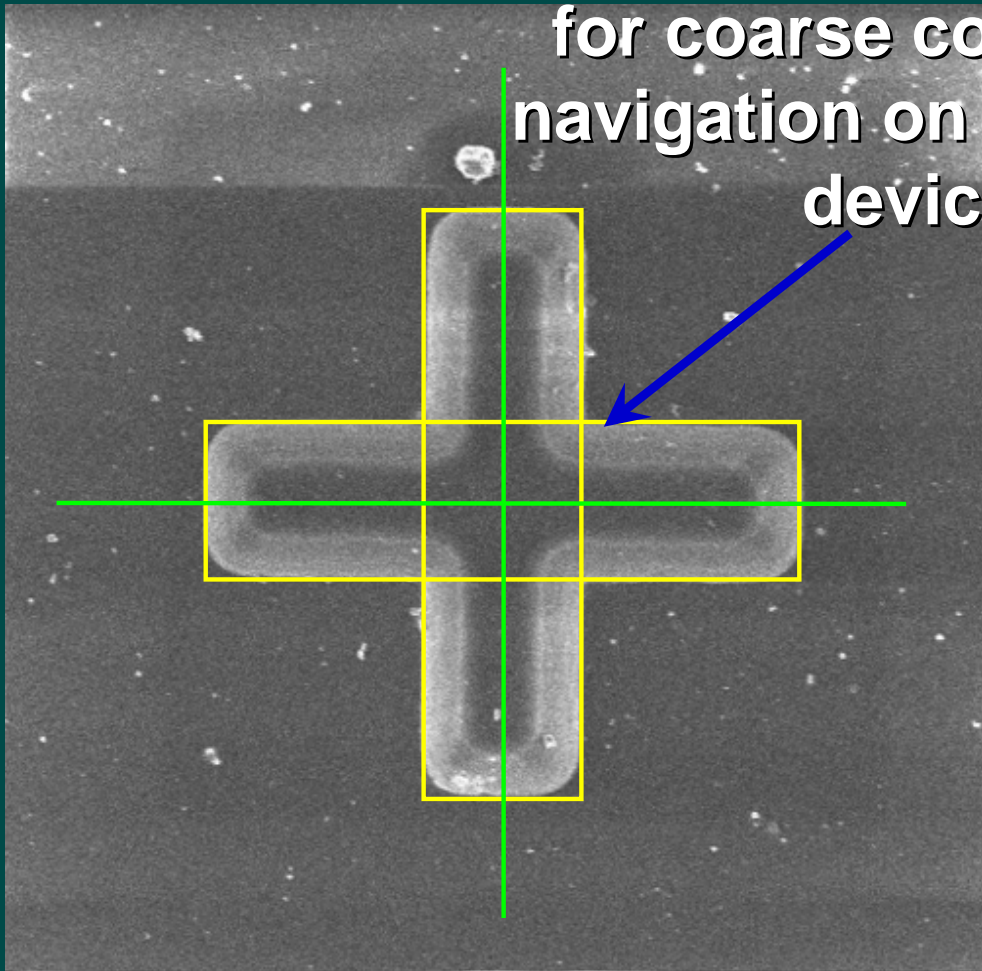


Targeted Node



References and Nodes in FIB

Use alignment references
for coarse coordinate
navigation on sacrificial
device



Then remove dielectric and
capture precise position of
node

Navigation with Local Alignment

- ◆ Accuracy of FIB stage is limited – how to navigate on small-linewidth devices?
- ◆ Use coordinates for coarse navigation
- ◆ Use protective shield as your local reference!
- ◆ Try to make contacts in between shield lines, if impossible then bypass the shield

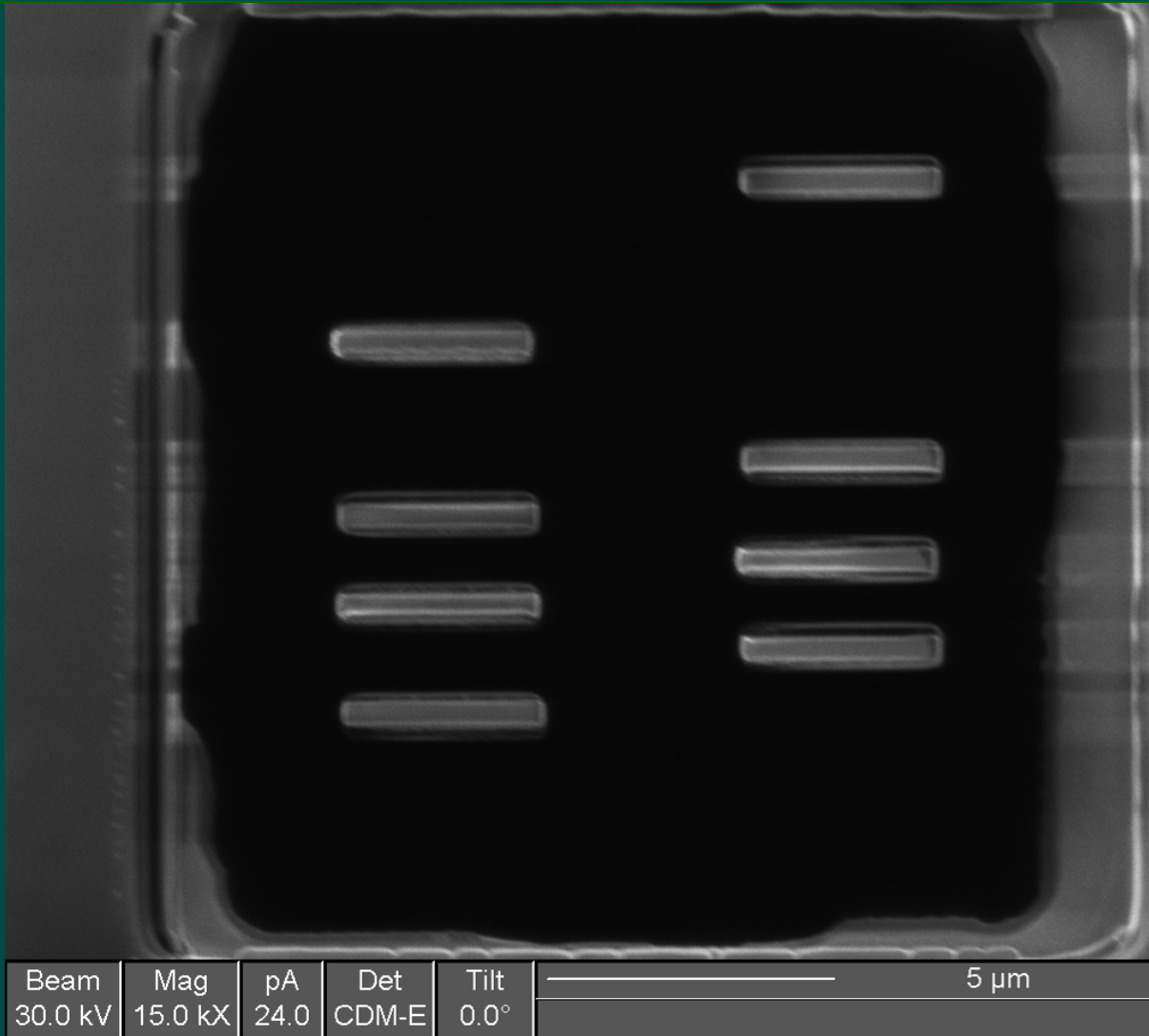
Bypassing Shield

- ◆ **Bypass entire shield**
 - » Best for analog shields, works for some digital
 - » Takes 30 to 60 min. of FIB time per device
 - » Often need follow-up by non-FIB techniques
 - » Shield can be removed to speed up further work
- ◆ **Bypass protective shield locally**
 - » Works on analog and digital shields
 - » On $<200\text{nm}$ devices 2 – 3 shield lines may need bypassing to clear space for each contact
 - » Takes 30 to 120 min. of FIB time per bypass

Disabling Shield

- ◆ **Disable shield control circuitry**
 - » Requires detailed analysis of layout
 - » Simulate “OK” shield on input of test circuitry
 - » Cut output of charge pump – disable flash erase!
 - » Cut “security interrupt” outputs, tie to “1” or “0”

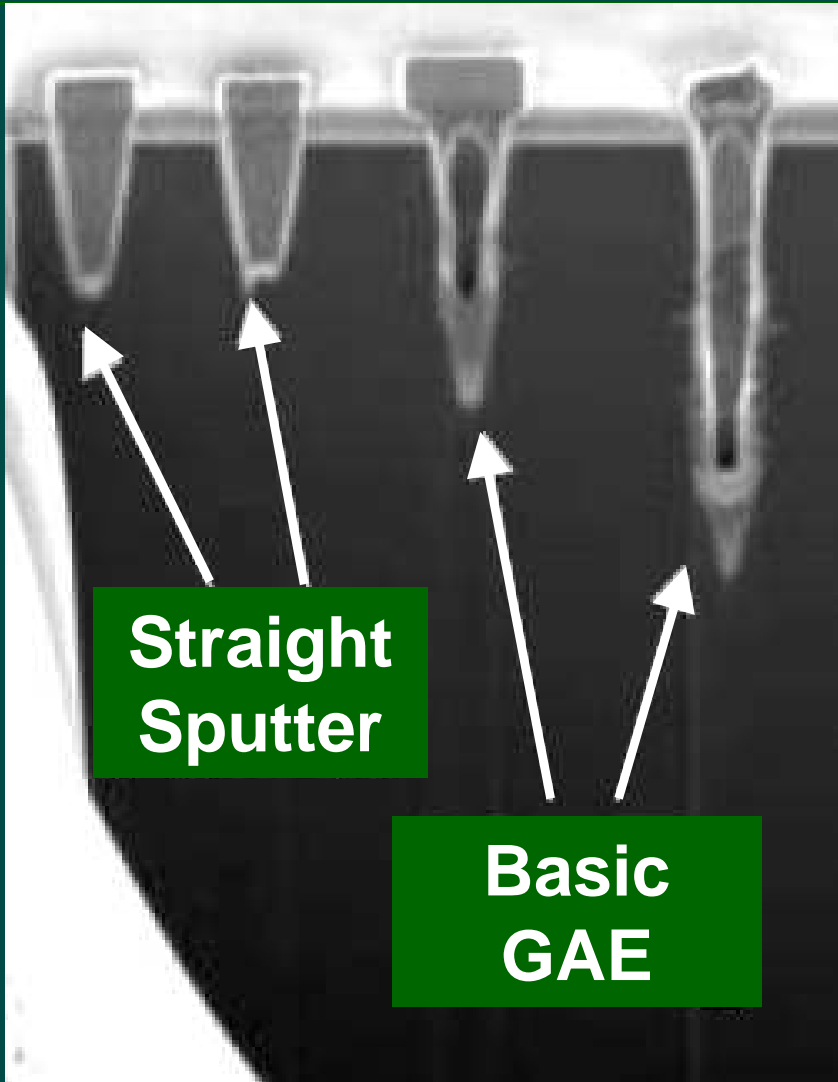
Making “Large” Contacts: Direct Line Probing



On devices with spacious layout and line-width $\geq 350\text{nm}$ direct probing of internal nodes may be possible.

FIB Image of bus opening
© Christopher Tarnovsky
www.flylogic.net

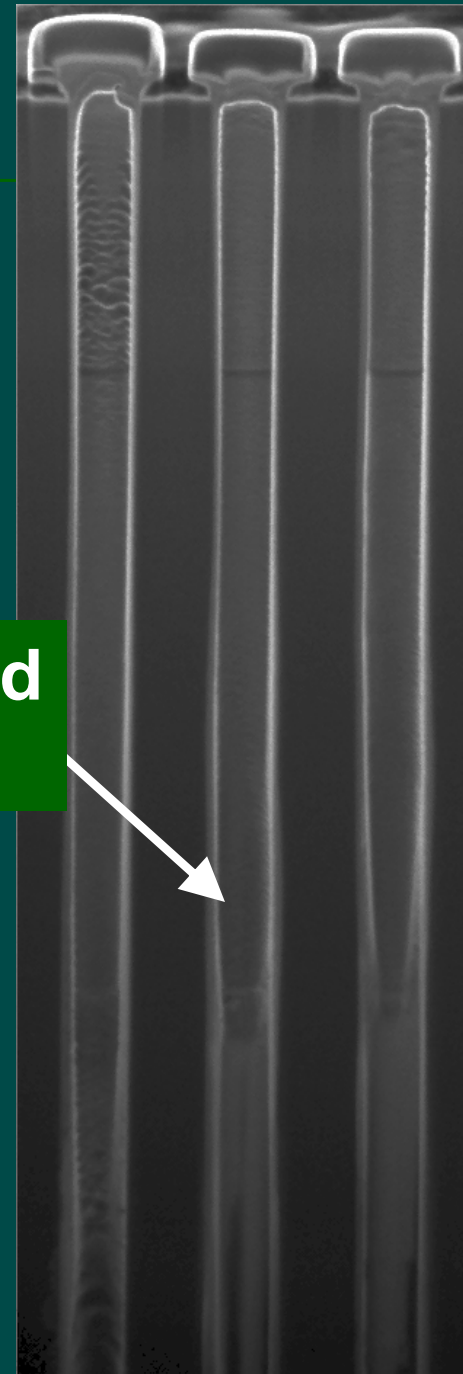
Making Small Contacts: High Aspect Ratio Milling



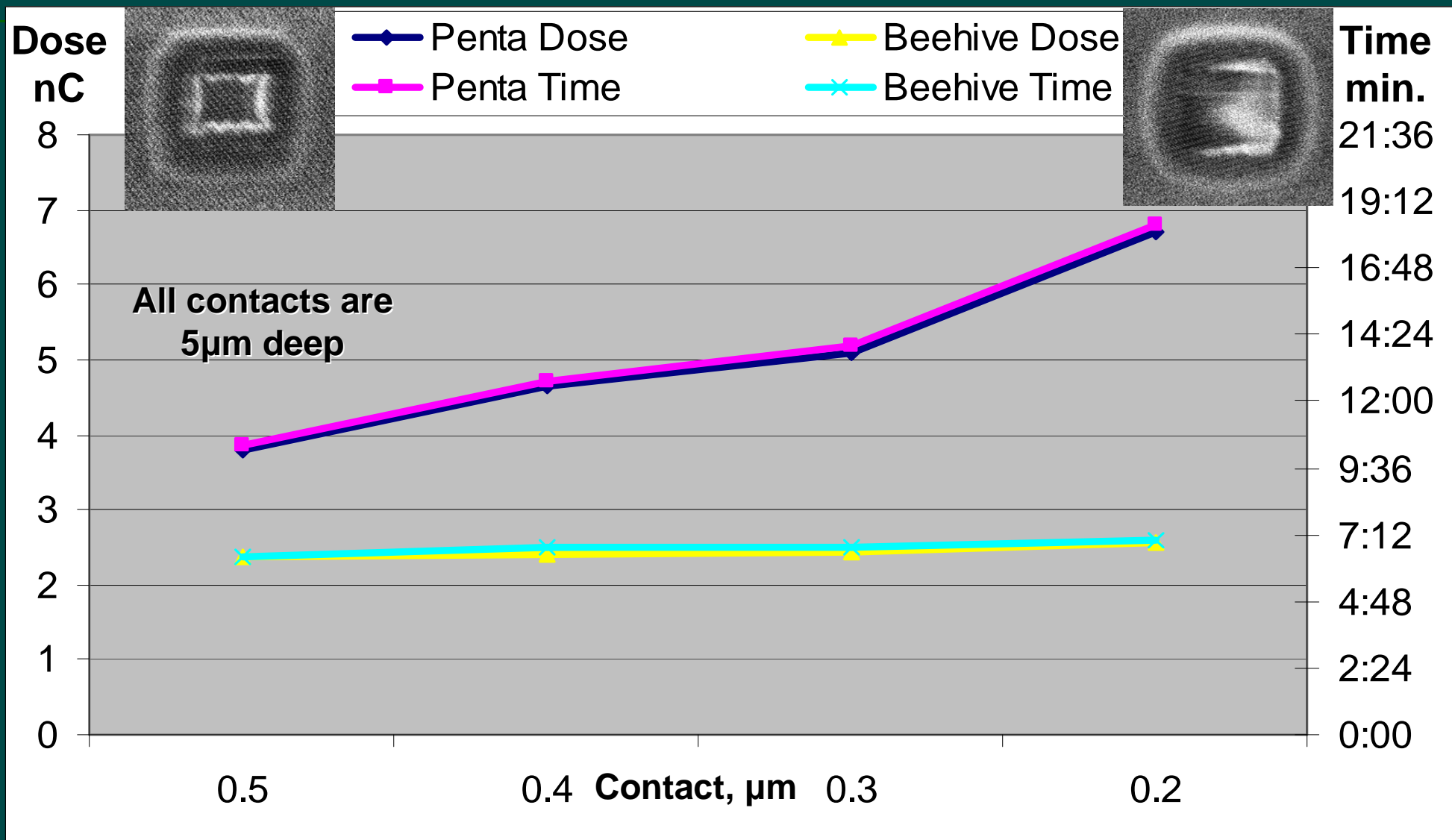
**Straight
Sputter**

**Basic
GAE**

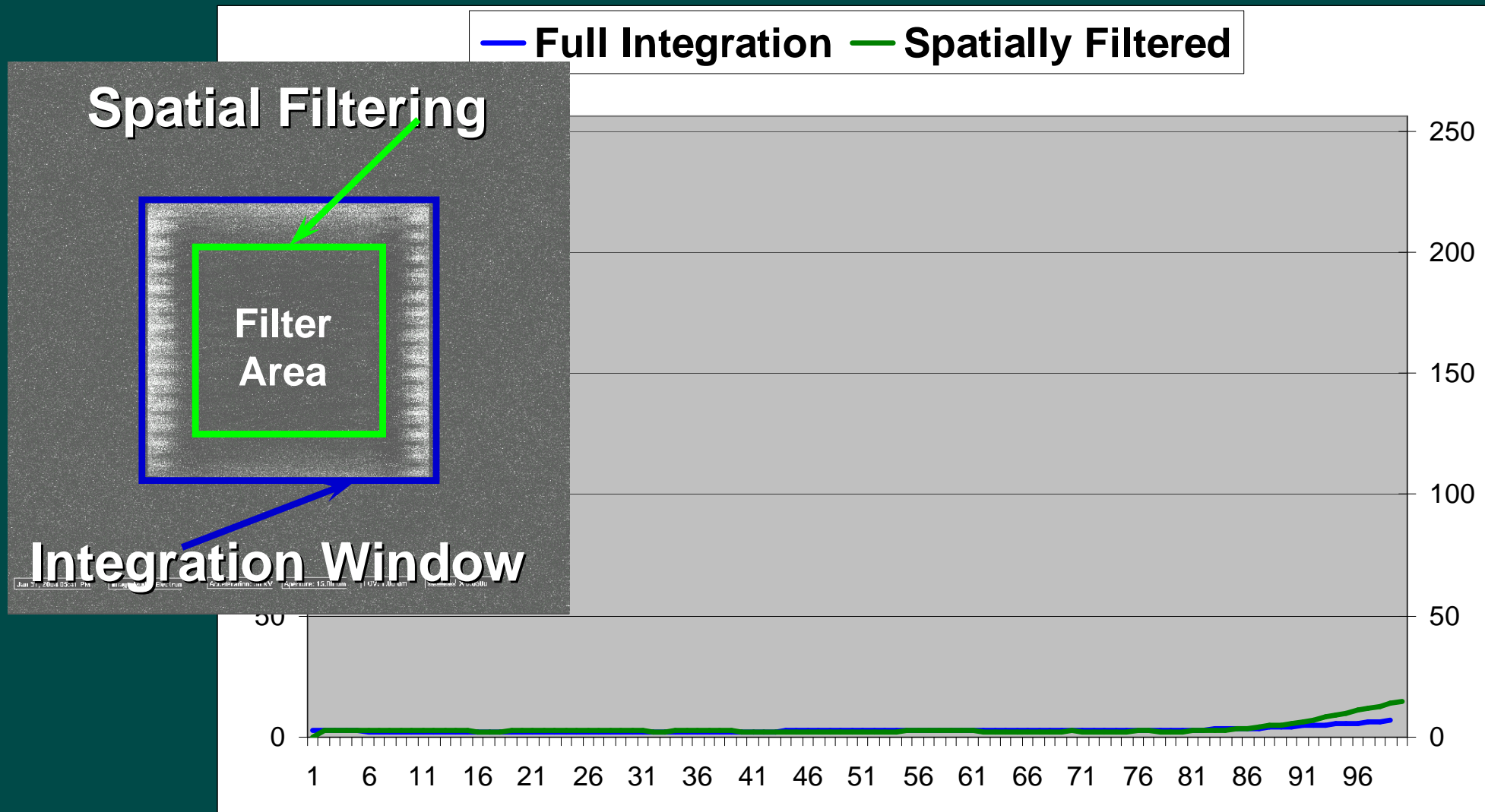
**Optimized
GAE**



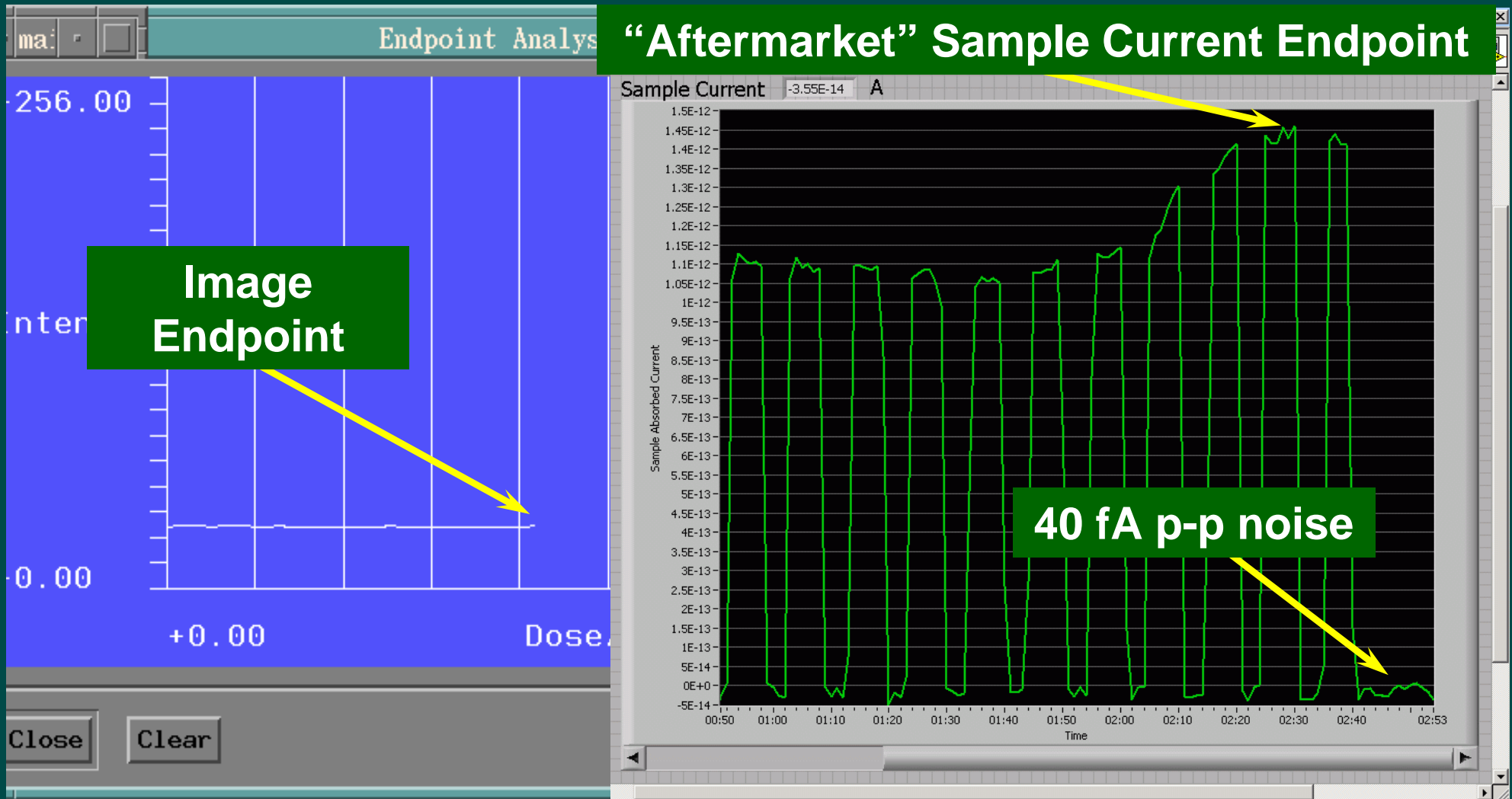
Making Small Contacts: High Aspect Ratio Milling Throughput



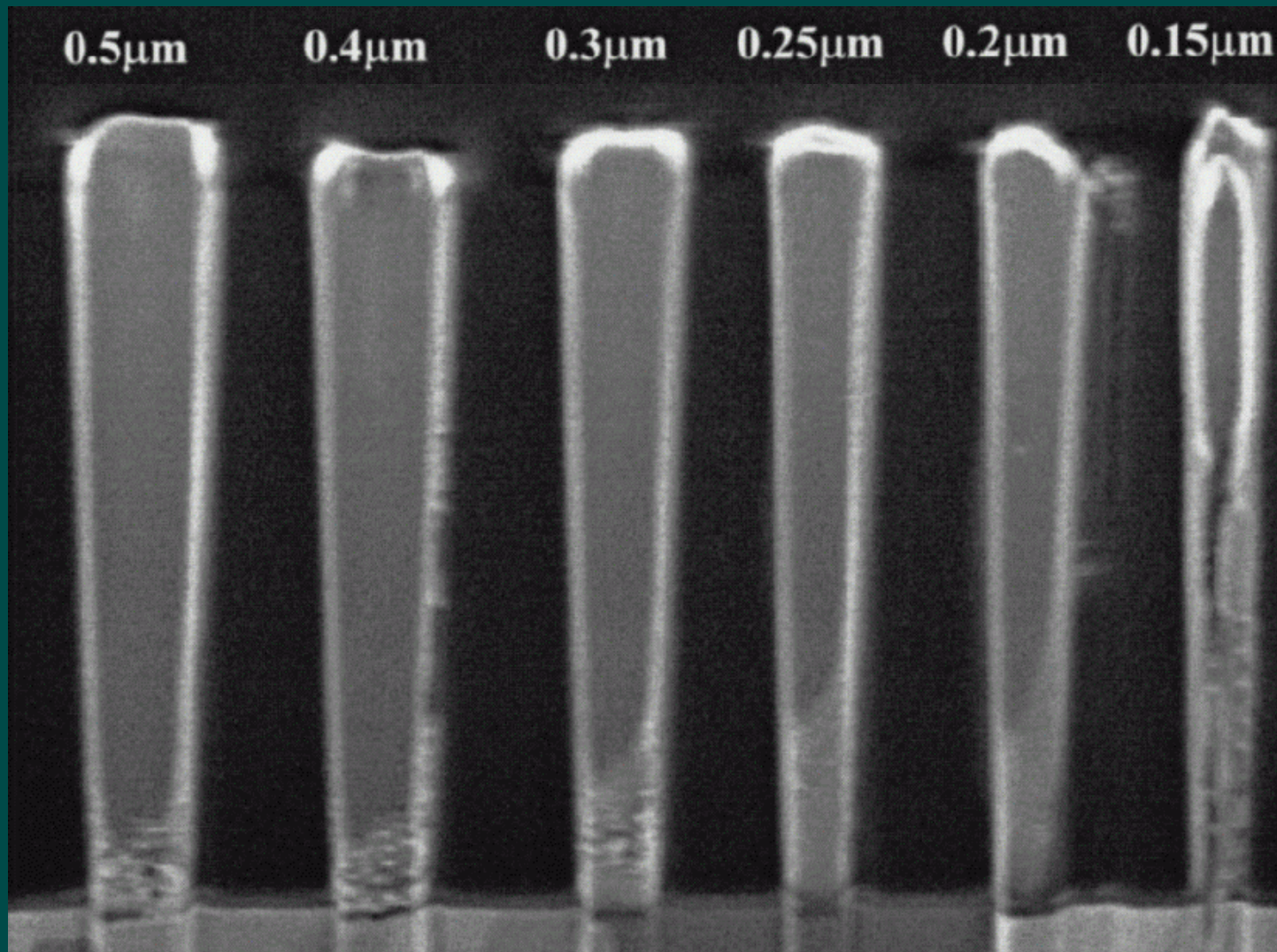
Making Small Contacts: High Aspect Ratio Endpointing by Image



Making Small Contacts: High Aspect Ratio Endpointing by Current



Making Small Contacts: High Aspect Ratio Deposition

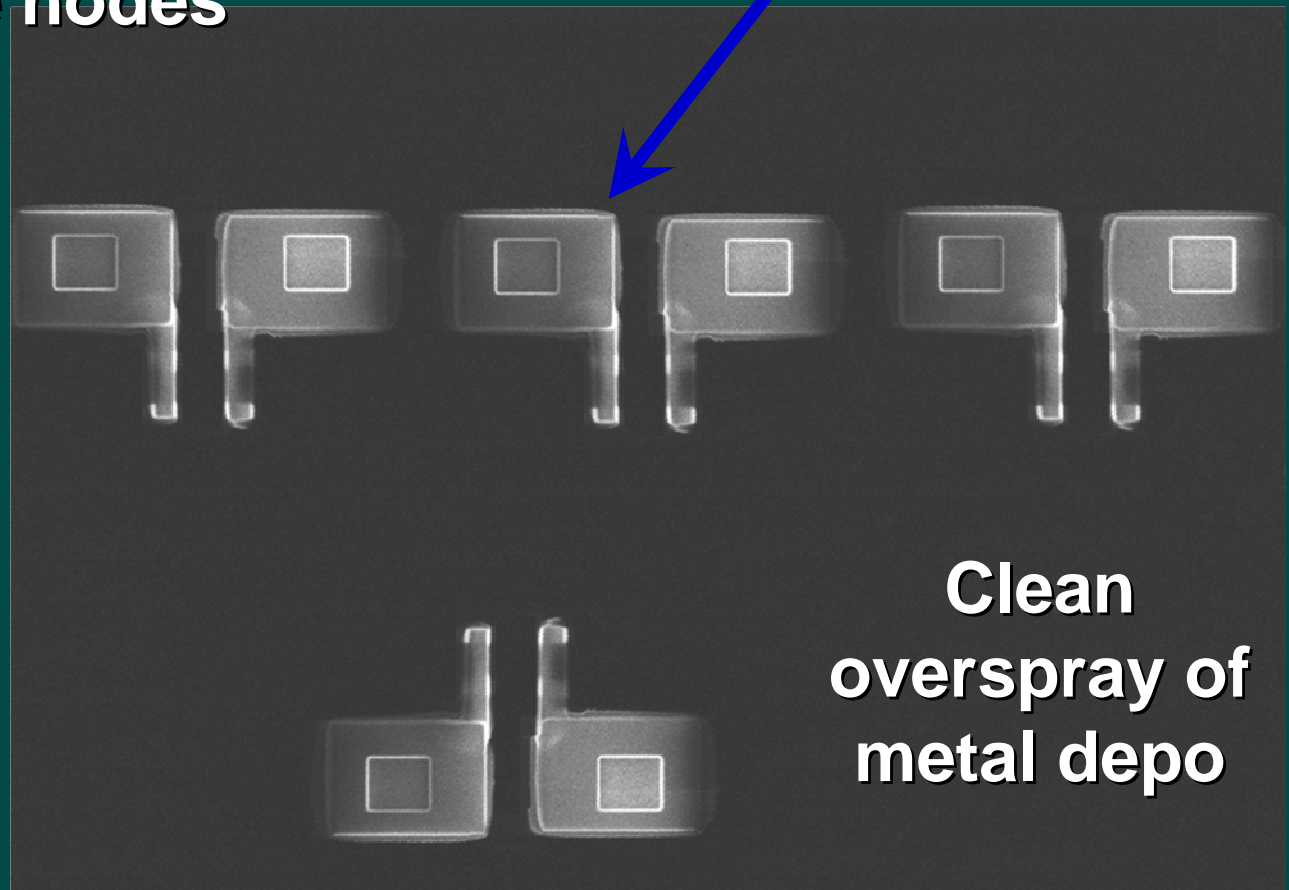


Making Contacts and Pads

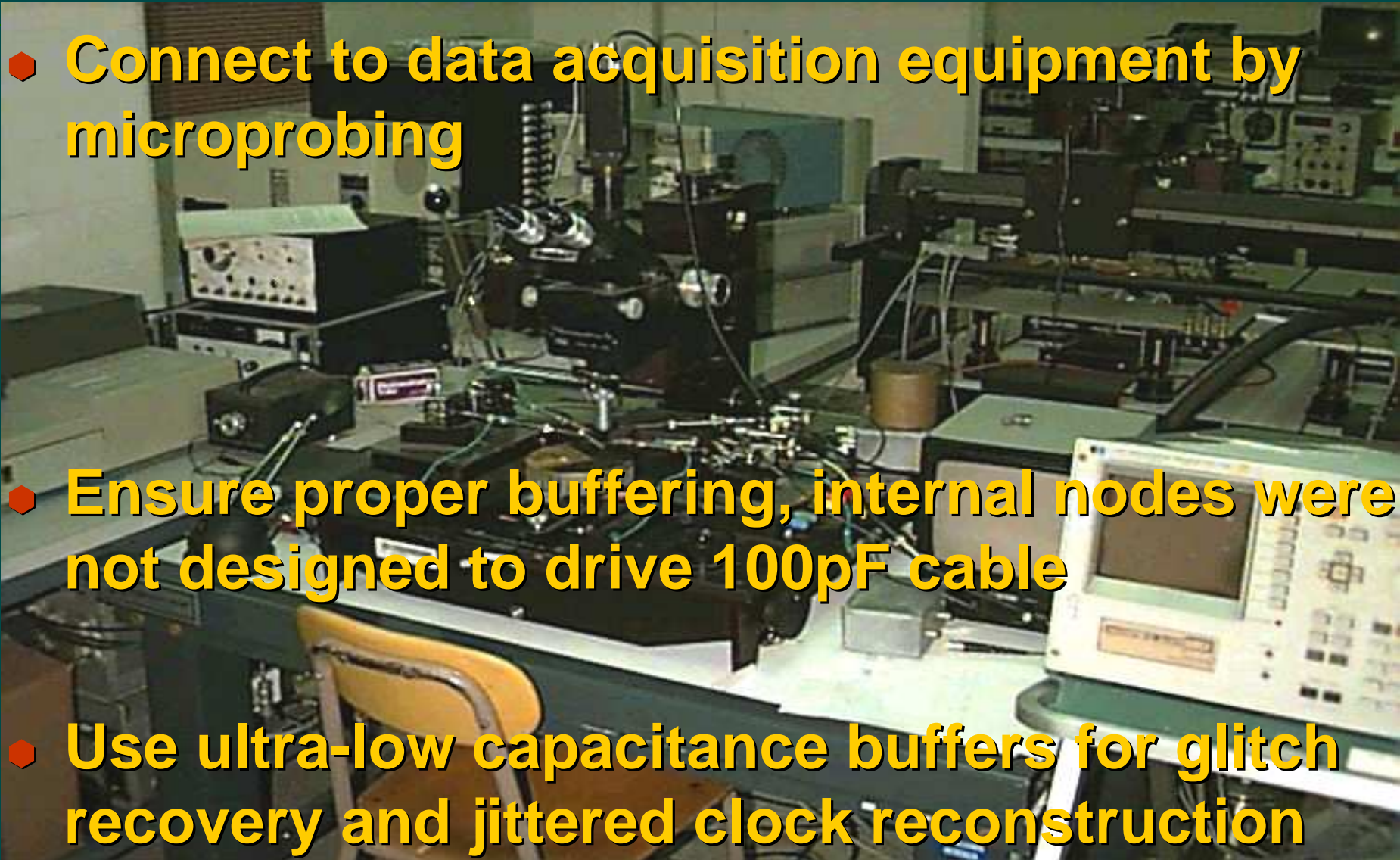


HAR via
connecting
to the nodes

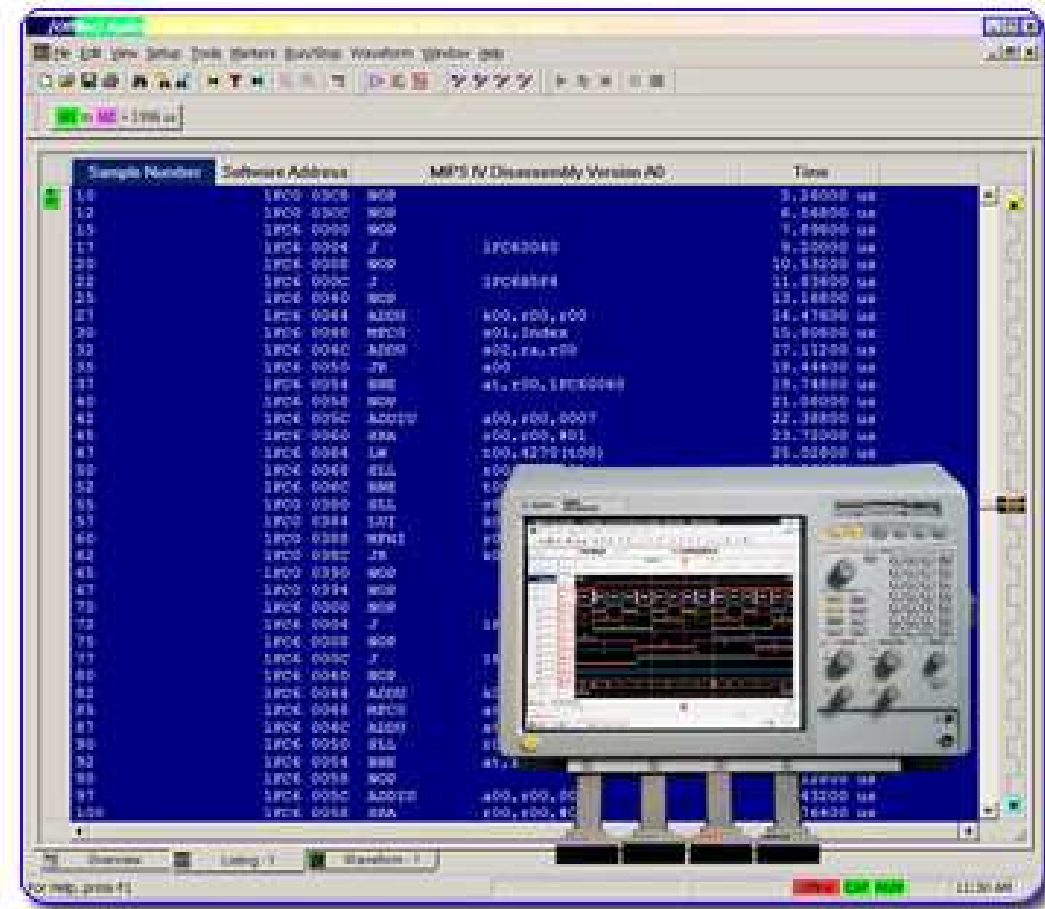
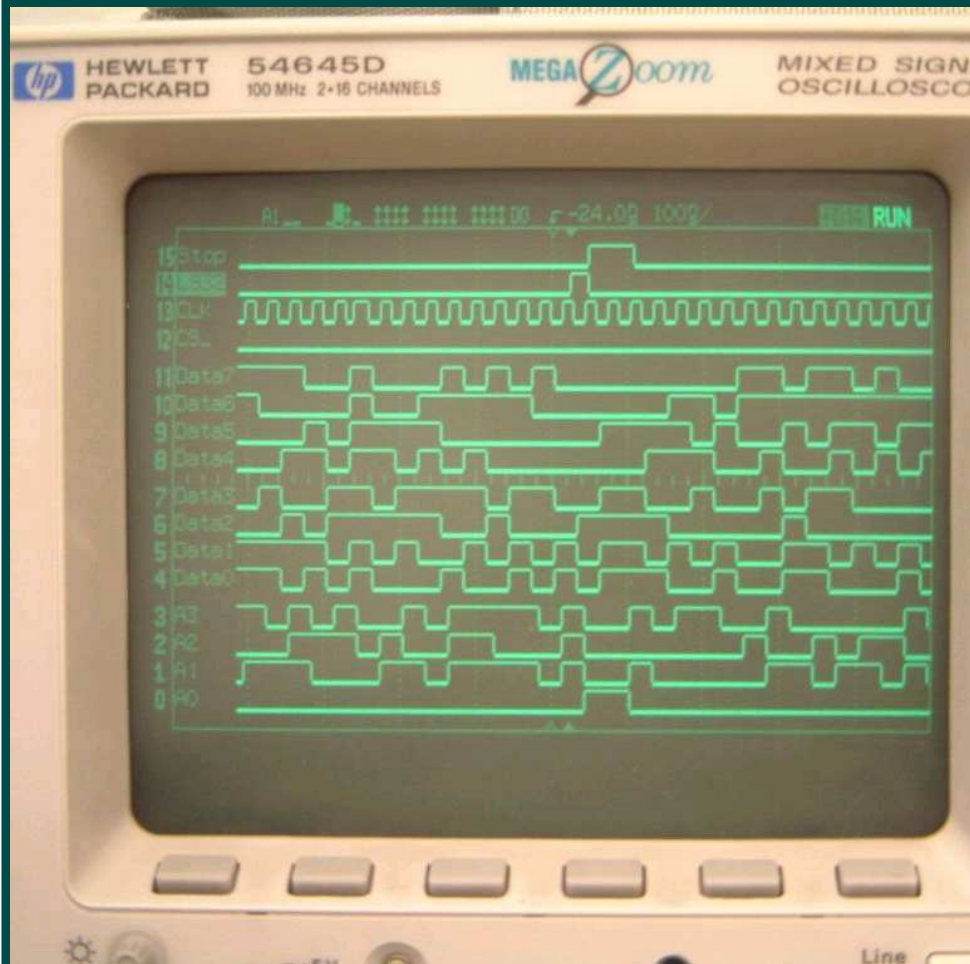
Contact pads for probing



Data Extraction

- ◆ Connect to data acquisition equipment by microprobing
 - ◆ Ensure proper buffering, internal nodes were not designed to drive 100pF cable
 - ◆ Use ultra-low capacitance buffers for glitch recovery and jittered clock reconstruction
- 

Filter, Align, and Convert to HEX Code



Disassemble, decompile, and make yourself at home ☺

The screenshot displays the IDA Pro disassembler interface. The main window shows assembly code for a function named 'main'. The code includes instructions like 'nr', 'subi', 'clrwi', 'li', 'stwu', 'luz', 'addi', 'slwi', 'add', 'lis', 'ori', 'ntctrl', 'bctrl', 'trap', 'nflr', and 'stwu'. The status bar at the bottom indicates 'AU: idle', 'Down', 'Disk: 26GB', '0000176C', and '0000276C: _text:0000276C'.

Names window:

Name	Address
__exit_stub	000091
__keymgr_dwarf2_register_sections_stub	000092
__atexit_stub	000092
__dyld_register_func_for_remove_image_stub	000092
__dyld_register_func_for_add_image_stub	000092
__init_keymgr_stub	000092
__free_stub	000092
__abort_stub	000092
__keymgr_set_and_unlock_processwide_ptr...	000092
__keymgr_get_and_lock_processwide_ptr_stub	000093
__calloc_stub	000093

Strings window:

Address	Length	Type	String
..."_cstring...	00000016	C	__dyld_mod_term_fur
..."_cstring...	0000002D	C	__dyld_make_delaye
..."_cstring...	00000013	C	__dyld_image_count
..."_cstring...	00000016	C	__dyld_get_image_nz
..."_cstring...	00000018	C	__dyld_get_image_he
..."_cstring...	0000001D	C	__dyld_NSLookupSym
..."_cstring...	00000019	C	__dyld_NSAddressOf
..."_cstring...	00000008	C	libobjc
..."_cstring...	0000000B	C	__objcIrit
..."_cstring...	0000004F	C	The kernel support fo
..."_cstring...	0000000A	C	/mach.sym

Executing Function 'main'...
Compiling file 'C:\Program Files\IDA Pro Advanced v4.5\idc\onload.idc'...
Executing Function 'Onload'...
IDA is analysing the input file...
You may start to explore the input file right now.
Propagating type information...
Function argument information is propagated
The initial autoanalysis is finished.

Signal injection

- Injection of impulses into data bus can alter execution of embedded code
- Basic application: disruption of “End Of Loop” command in Answer To Reset (ATR) function of smartcard microcontrollers may cause extraction of data memory 😊
- Suitable injection buffers are not available from OEMs of pattern generators or otherwise, must DIY

Limitations of existing FIB technology

- ◆ Accuracy of navigation
 - » Targeting nodes by only the coordinates on devices with linewidth $< 200\text{nm}$ is unreliable on most FIBs
- ◆ Aspect ratio of contacts
 - » Endpoint of milling on contacts with 20:1 depth/width may require “aftermarket” upgrades
- ◆ Linewidth (technology node) limitations
 - » Making multiple deep contacts smaller than 250 nm is esoteric art, takes very dedicated operator...

Countermeasures against FIB

- FIB attacks can be made uneconomical for casual FIB access:
 - » Planarise devices and use small linewidth technology
 - » Thick copper plane combined with active shields is difficult to cut
 - » Use Liquid Crystal Polymer for encapsulation – hard to remove
 - » Use combination of analog and digital shields: bypass is difficult
 - » Introduce “position jitter” to shield layer on lithography step to prevent local referencing for navigation
 - » Orient straight shield lines at 45 degrees angle to the layout

Summary

- **FREUD by FIB methods can't be prevented, but can be made too expensive for casual access**
- **Basic countermeasures are relatively simple in manufacturing – planarisation of devices, more sophisticated active shields**
- **Advanced countermeasures become viable as cost of IC manufacturing is reduced: active double-shielding, LCP encapsulation, 45 degrees shield-to-layout orientation and shield position “jittering”**

www.partbeamsystech.com

FIB User Group, Washington DC, USA