

How Do I Crack Satellite and Cable Pay TV?



Chris Gerlinsky

 @akacastor

chris@pdrnorth.com

Attacking the Digicipher 2 conditional access system used in millions of TV set-top-boxes in North America.



REcon 0xA June 17-19 2016

Digital Television

Satellite

Cable

Modulation:

DC2 QPSK ~27 Mbit

8PSK Turbo FEC ~38 Mbit

Modulation:

QAM 256 ~38 Mbit

Out-of-band:

QPSK 2 Mbit

Video format:

MPEG-2 or H.264 Transport Stream (MPEG TS)

Encryption:

Digicipher 2 (not DVB standard)

MPEG Transport Stream

27 – 38 Mbit

188-byte packets

Categorized by 13-bit **PID**
(0 - 0x1FFF)

MPEG PES

Packetized Elementary Streams
64 Kbit – 19 Mbit
Video, audio

Service Information Tables

8-bit table ID (0 – 0xFF)
Up to 1024 bytes, with CRC32

Table 0x00 – (PID 0) Program Association Table
Table 0x01 – (PID 1) Conditional Access Table
Table 0x02 – Program Map Table

PAT contains list of programs: PID carrying PMT
PMT contains list of PIDs for video, audio, ECM

Table 0x40 – ECM40
Table 0x41 – ECM41
Table 0x95 – EMM95

ECM are sent in pairs
Cable: EMM are OOB

Genpix SkyWalker-1 USB satellite interface



<http://updatelee.blogspot.ca/2010/09/genpix-skywalker-1-linux-driver-mods.html>

<https://bitbucket.org/updatelee/v4l-updatelee>

Hauppauge HVR 950Q ATSC / QAM USB interface



Using dvbsnoop to view PMT PID 0x129

```
$ dvbsnoop -if log-11959.ts -s ts -tssubdecode 0x129
```

```
dvbsnoop V1.4.50 -- http://dvbsnoop.sourceforge.net/
```

```
-----  
TS-Packet: 00000001 PID: 297 (0x0129), Length: 188 (0x00bc)  
from file: 2016-05-21-full-log-11959.ts  
-----
```

```
0000: 47 61 29 1b 00 02 80 26 00 0f 15 00 00 01 10 00 Ga)...&.....  
0010: 06 09 04 47 49 01 29 80 01 10 00 03 83 01 00 81 ..GI.).....  
0020: 01 11 00 06 0a 04 65 6e 67 00 6c c4 86 25 40 00 ...eng.l...%@.  
0030: 11 00 3b 40 0f 2a 63 4a 22 04 00 00 02 0a cf 4f ;;@.*cJ".....0  
0040: fe 3b 41 40 31 10 20 00 3b 40 0f 75 e5 22 27 3b ;;A@1. ;;@.u."";  
0050: 09 e3 40 85 71 da 6c fe 7f e6 ca 00 00 00 08 01 ..@.q.l.....Y.  
0060: 00 01 59 03 00 00 72 03 01 00 d7 2b 46 ee d4 bf ..Y...r....+F...  
0070: 6c c5 06 d2 2d 09 ff ff ff ff ff ff ff ff ff ff l.....  
0080: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff .....  
0090: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff .....  
00a0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff .....  
00b0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff .....
```

```
Sync-Byte 0x47: 71 (0x47)  
Transport_error_indicator: 0 (0x00) [= packet ok]  
Payload_unit_start_indicator: 1 (0x01) [= Packet data starts]  
transport_priority: 1 (0x01)  
PID: 297 (0x0129) [= ]  
transport_scrambling_control: 0 (0x00) [= No scrambling of TS packet payload]  
adaptation_field_control: 1 (0x01) [= no adaptation_field, payload only]  
continuity_counter: 11 (0x0b) [= (sequence ok)]
```

```
    Payload: (len: 184)  
    ==> pointer_field: 0 (0x00)  
    ==> Section table: 2 (0x02) [= Program Map Table (PMT)]  
Data-Bytes:  
0000: 00 02 80 26 00 0f 15 00 00 01 10 00 06 09 04 47 ...&.....G  
0010: 49 01 29 80 01 10 00 03 83 01 00 81 01 11 00 06 I.).....  
0020: 0a 04 65 6e 67 00 6c c4 86 25 40 00 11 00 3b 40 ..eng.l...%@;  
0030: 0f 2a 63 4a 22 04 00 00 02 0a cf 4f fe 3b 41 40 .*cJ".....0;A@  
0040: 31 10 20 00 3b 40 0f 75 e5 22 27 3b 09 e3 40 85 1. ;;@.u."";;.@.  
0050: 71 da 6c fe 7f e6 ca 00 00 00 08 01 00 01 59 03 q.l.....Y.  
0060: 00 00 72 03 01 00 d7 2b 46 ee d4 bf 6c c5 06 d2 ..r....+F...l...  
0070: 2d 09 ff ff ff ff ff ff ff ff ff ff ff ff ff ff .....  
0080: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff .....  
0090: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff .....  
00a0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff .....  
00b0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff .....
```

```
TS sub-decoding (1 packet(s) stored for PID 0x0129):  
=====
```

```
TS contains Section...  
SI packet (length=41):  
  PID: 297 (0x0129)  
  
Guess table from table id...  
PMT-decoding...  
Table_ID: 2 (0x02) [= Program Map Table (PMT)]  
section_syntax_indicator: 1 (0x01)  
(fixed '0'): 0 (0x00)  
reserved_1: 0 (0x00)  
Section_length: 38 (0x0026)  
Program_number: 15 (0x000f)  
reserved_2: 0 (0x00)  
Version_number: 10 (0x0a)  
current_next_indicator: 1 (0x01) [= valid now]  
Section_number: 0 (0x00)  
Last_Section_number: 0 (0x00)  
reserved_3: 0 (0x00)  
PCR_PID: 272 (0x0110)  
reserved_4: 0 (0x00)  
Program_info_length: 6 (0x0006)
```

```
  MPEG-DescriptorTag: 9 (0x09) [= CA_descriptor]  
  descriptor_length: 4 (0x04)  
  CA_system_ID: 18249 (0x4749) [= General Instrument]  
  reserved: 0 (0x00)  
  CA_PID: 297 (0x0129)
```

```
Stream_type loop:  
  
Stream_type: 128 (0x80) [= User private]  
reserved_1: 0 (0x00)  
Elementary_PID: 272 (0x0110)  
reserved_2: 0 (0x00)  
ES_info_length: 3 (0x0003)
```

```
  DVB-DescriptorTag: 131 (0x83) [= User defined/ATSC reserved]  
  descriptor_length: 1 (0x01)  
  Descriptor-data:  
    0000: 00
```

```
Stream_type: 129 (0x81) [= User private]  
reserved_1: 0 (0x00)  
Elementary_PID: 273 (0x0111)  
reserved_2: 0 (0x00)  
ES_info_length: 6 (0x0006)
```

```
  MPEG-DescriptorTag: 10 (0x0a) [= ISO_639_language_descriptor]  
  descriptor_length: 4 (0x04)  
  ISO639_language_code: eng  
  Audio_type: 0 (0x00) [= undefined]
```

```
CRC: 1824818725 (0x6cc48625)
```

DC2 Service Information Tables

SCTE 65: Service Information Delivered Out-Of-Band For Digital Cable Television

DC2 tables

VCT – Virtual Channel Table
NIT – Network Information Table
NTT – Network Text Table
MGT – Master Guide Table

Many DVB standard tables are not used:

DVB equivalent

BAT – Bouquet Association Table
SDT – Service Description Table
EIT – Event Information Table



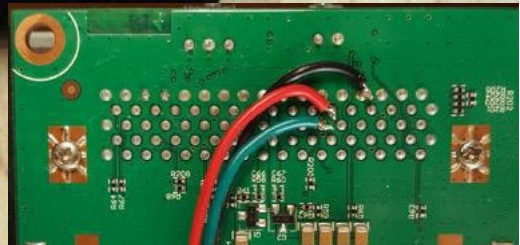
ENGINEERING COMMITTEE
Digital Video Subcommittee

AMERICAN NATIONAL STANDARD

ANSI/SCTE 65 2016

SERVICE INFORMATION
DELIVERED OUT-OF-BAND FOR
DIGITAL CABLE TELEVISION

www.scte.org/SCTE/Standards/Download/SCTE/Standards/Download_SCTE_Standards.aspx?hkey=63914a25-0f85-4607...
Society of Cable Telecommunications Engineers | International Society of Broadband Experts
ABOUT EDUCATION EVENTS RESOURCES MEMBERSHIP AREAS OF INTEREST STANDARDS CERTIFICATION
CHAPTERS JOIN CONTACT
Home > Standards > Download SCTE Standards
Download SCTE Standards
The following documents are consensus developed and SCTE Engineering Committee approved SCTE standards. Many of these standards have been approved by the American National Standards Institute (ANSI) and are so designated.
Download SCTE-Approved Standards
Electronic copies of approved SCTE standards are complimentary and available free of charge. Click on the sections and titles of the standards below to download in PDF format.
JOIN SCTE STANDARDS
Standards
Download SCTE Standards
Recommended and Operational Practices
Standards Member Directory
Engineering Committee
Standards Categories
Standards Bulletin
Standards FAQs
Procedures and Forms
SCTE Standards Workspace
Join the Standards Program
ANSISCTE 01.2015 Specification for "F" Port, Female, Outdoor
ANSISCTE 02.2015 Specification for "F" Port, Female, Indoor
ANSISCTE 03.2009 Test Method for Coaxial Cable Structural Return Loss
ANSISCTE 04.2014 Test Method for "F" Connector Return Loss
ANSISCTE 05.2014 Test Method for "F" Connector Return Loss In-Line Pair
ANSISCTE 06.2013 Composite Distortion Measurements (CSO & CTB)
ANSISCTE 07.2013 Digital Transmission Standard For Cable Television
ANSISCTE 08.2010 Test Method for Coax Bend
ANSISCTE 10.2014 Test Method for Flexible Coaxial Cable Impact
ANSISCTE 11.2012 Test Method for Aerial Cable Corrosion Protection Flow
ANSISCTE 12.2011 Test Method for Center Conductor Bond to Dielectric for Trunk, Feeder and Distribution Coaxial Cables
ANSISCTE 13.2011 Dielectric Air Leakage Test Method For Trunk, Feeder and Distribution Coaxial Cable
ANSISCTE 14.2011 Test Method for Hex Crimp Tool Verification/Calibration
ANSISCTE 15.2006 Specification for Trunk, Feeder and Distribution Coaxial Cable
ANSISCTE 16.2012 Test Procedure for Hum Modulation
ANSISCTE 17.2007 Test Procedure for Carrier to Noise (CN, CCN, C/N, CTN)
SCTE 18.2013 Emergency Alert Messaging for Cable
(ANSI J-STD-426-2013)
ANSISCTE 19.2013 Methods for Synchronous Data Services Transport
ANSISCTE 20.2012 Method for Carriage of Closed Captions and Non-Real Time Sampled Video
ANSISCTE 21.2012 Standard For Carriage Of NTSC VBI Data In Cable Digital Transport Streams
ANSISCTE 22-1.2012 Data-Over-Cable Service Interface Specification DOCSIS 1.0 Radio Frequency Interface (RFI)
ANSISCTE 22-2.2012 DOCSIS 1.0 Baseline Privacy Interface (BPI)
ANSISCTE 22-3.2012 DOCSIS 1.0 Operations Support System Interface (OSS)
ANSISCTE 23-1.2010 DOCSIS 1.1 Part 1: Radio Frequency Interface
ANSISCTE 23-2.2012 DOCSIS 1.1 Part 2: Baseline Privacy Plus Interface
ANSISCTE 23-3.2010 DOCSIS 1.1 Part 3: Operations Support System Interface
ANSISCTE 24-1.2009 IP-Cablecom 1.0 Part 1: Architecture Framework for the Delivery of Time Critical Services Over Cable Television Networks Using Cable Modems



SCTE 55-1 Decoding Software

https://www.scte.org/documents/pdf/Standards/ANSI_SCTE-55-1-2009.pdf



**Society of Cable
Telecommunications
Engineers**

ENGINEERING COMMITTEE
Digital Video Subcommittee

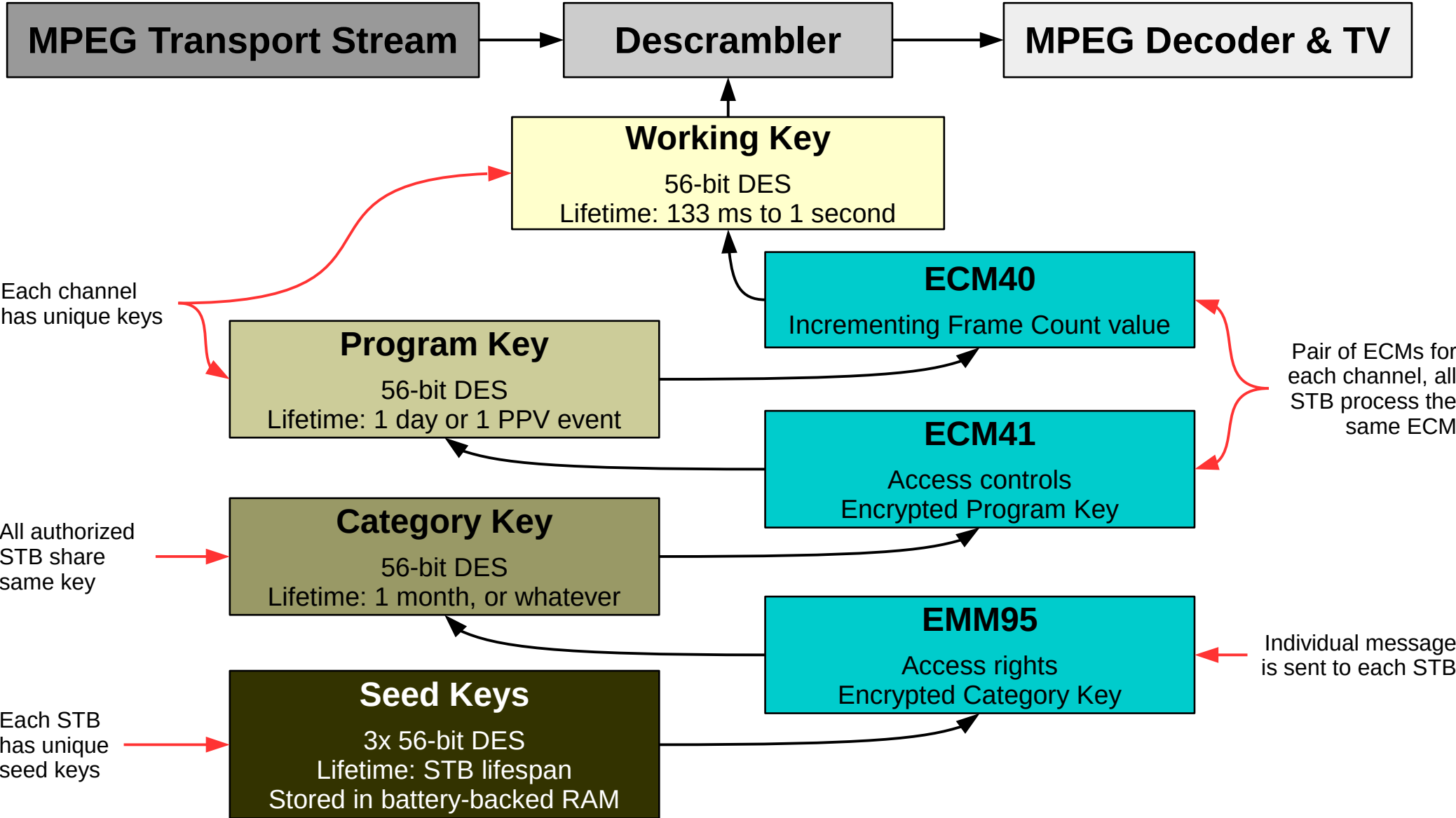
AMERICAN NATIONAL STANDARD

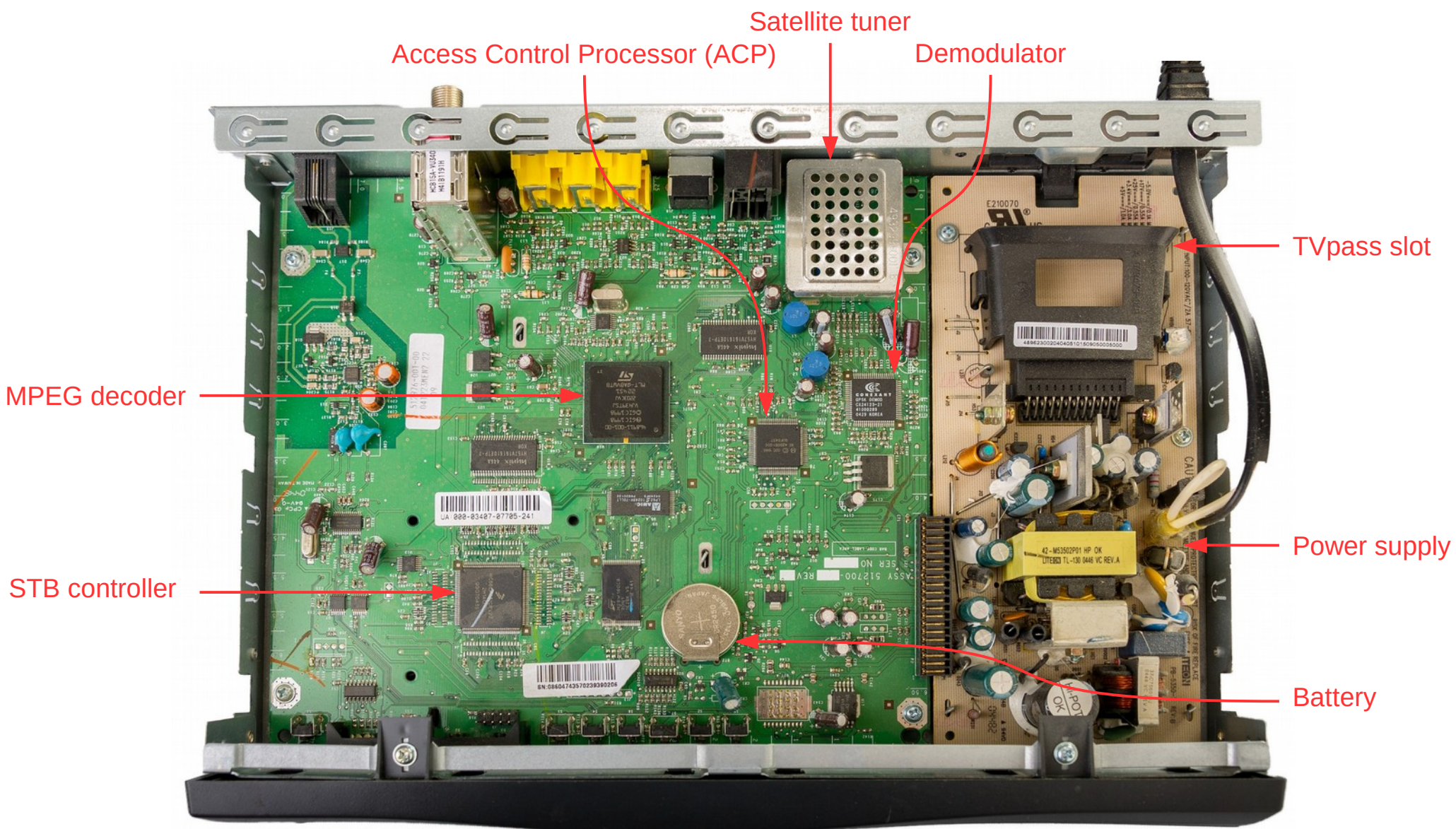
ANSI/SCTE 55-1 2009

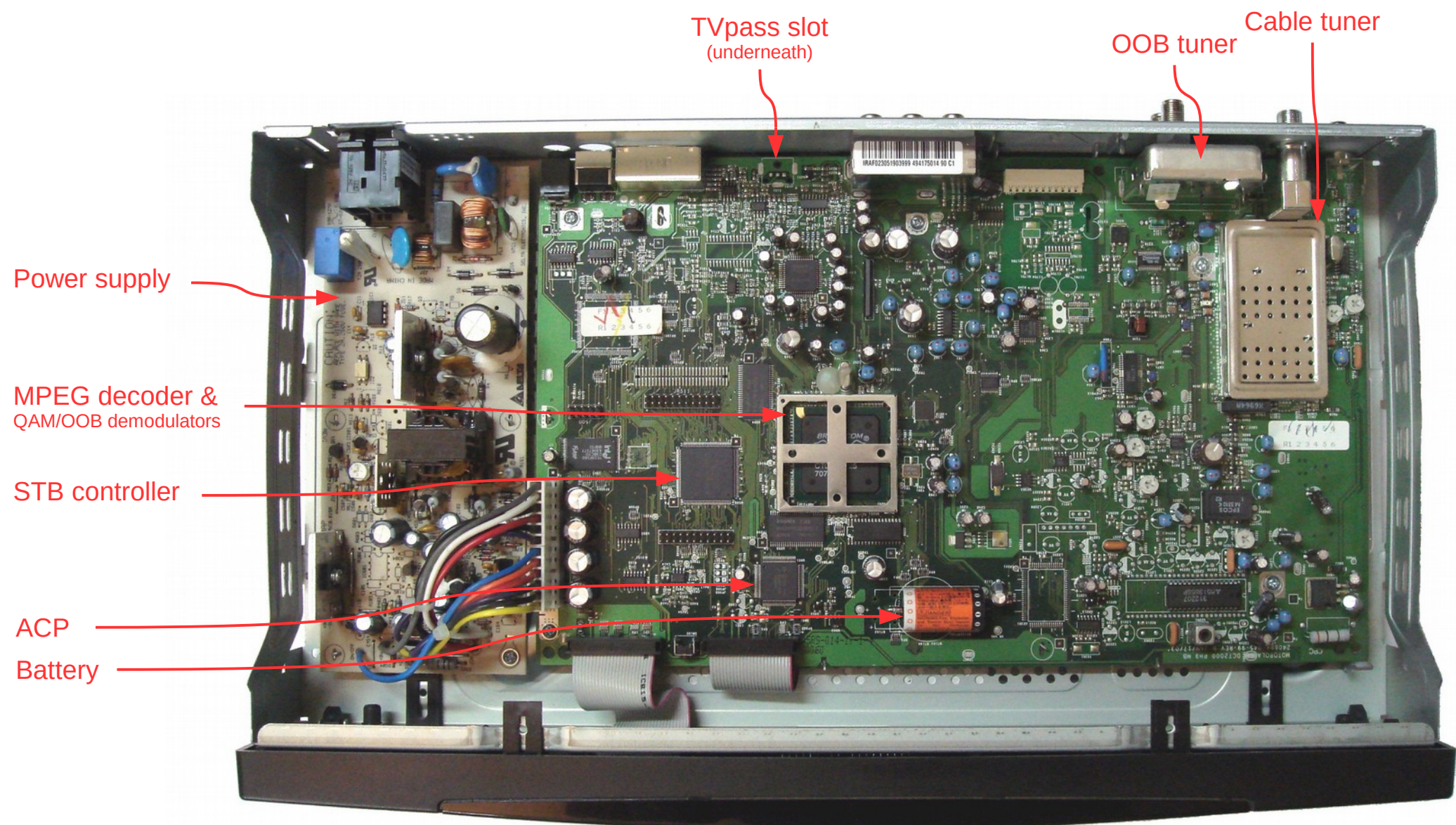
Digital Broadband Delivery System:
Out of Band Transport Part 1: Mode A

<https://github.com/akacastor/noob>
<https://github.com/akacastor/oobin>

```
283 //-----
284 // The process going from QPSK demodulator to TS data:
285 //-----
286
287     for( i=0; i<383-len; i+=384 )
288     {
289         // 0. Synchronize bitstream (find 0x47 0x64 0x47 0x64 ... sequence)
290
291         i += oob_synchronize_bitstream( data, i, len );
292         if( i+384>768-len )
293             break; // didn't synchronize before end of the bitstream
294
295         // data[0] is a 0x47 sync byte, data[192] is a 0x64 sync byte, there are two packets (384 bytes) to process
296
297         // 1. De-interleaver - run it twice (96 bytes x 2) to de-interleave a full ts packet
298
299         // works over 8 * 96-byte blocks, returns a single 96-byte assembled block
300         // data_in[] must contain at least 768 bytes
301         // return value: 0 if successful
302         for( n=0; n<4; n++ )
303         {
304             oob_de_interleaver( data+i + n*96, data_work + n*96 );
305             memcpy( data+i + n*96, data_work + n*96, 96 );
306         }
307
308         // 2. Reed Solomon Decoder - run it twice (96 bytes x 2) to fec a full ts packet, 4 times for a 384-byte block
309         // works over 96-byte blocks (runs twice for each ts packet)
310         // return value: 0 if successful - this 96-byte block is valid
311
312         if( do_fec )
313         {
314             for( n=0; n<4; n++ )
315                 fec_error[n] = oob_de_fec( data+i + n*96 );
316         }
317
318
319         // 3. Derandomizer - run it over ts packet - need to track even/odd sequence to be able to decrypt next packet
320         // OR
321         // - work on pairs of ts packets
322
323         // works over 384-byte blocks (2x ts packet)
324         // len is number of bytes to de randomize - usually this is 384, but less is accepted
325         // frame_pos is the position within 384-byte randomizer frame (ie: 192 if de_randomizing 2nd ts packet alone)
326         // return value: 0 if successful
327         oob_de_randomizer( data+i, 384, 0 );
328
329
330         if( do_fec )
331         {
332             for( n=0; n<2; n++ ) // loop through 2 TS packets to set TS error indicator if necessary
333             {
334                 if( fec_error[n*2] < 0 || fec_error[n*2 + 1] < 0 )
335
```









(M) (C) GIC 1998
XC 420061-006
HTF0203

C59
C54
R29
C58
75
51

C251

U8

50

C57
76

R53

C77

C60

C48

C152

R73
510

R61
0EE

C79

C47

C5

26

R63
330

C80

C55

C45

R70
100

C49

R79
E01

202

E01

R74
510

R93

C64

R62
0EE

R171

202

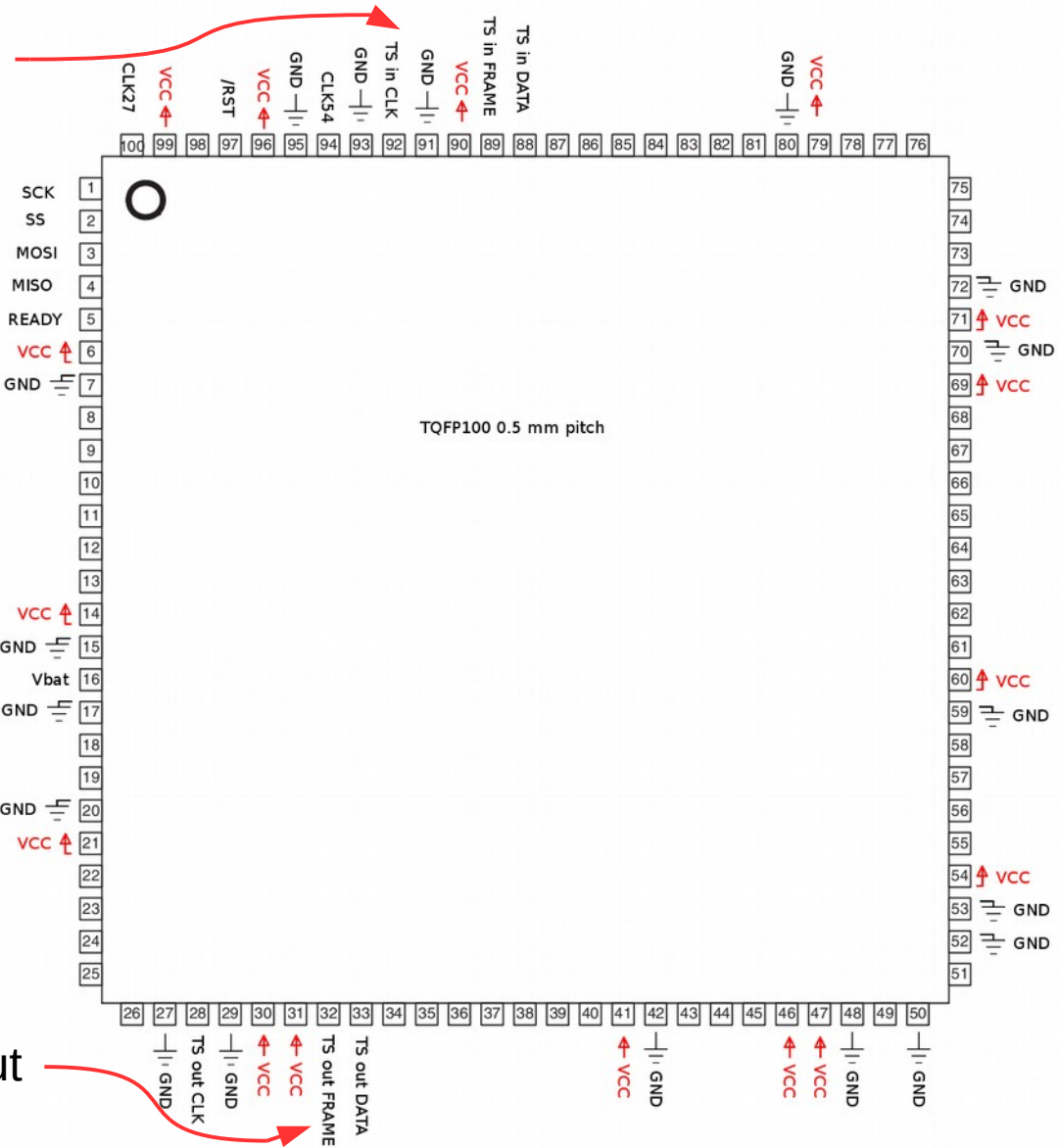
610

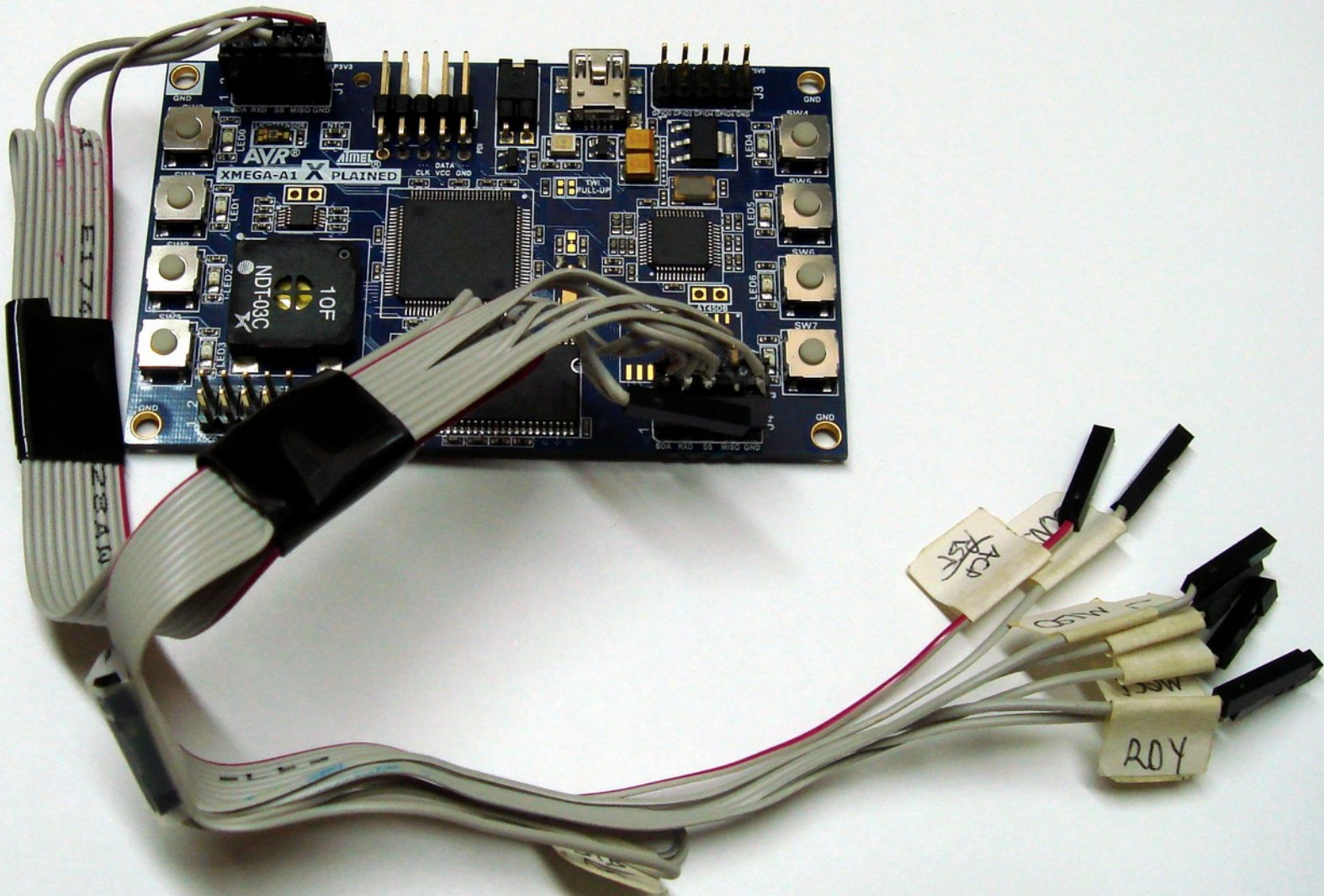
Transport Stream input
(from tuner)

SPI slave

Battery

Transport Stream output
(to MPEG decoder)





SPI data log

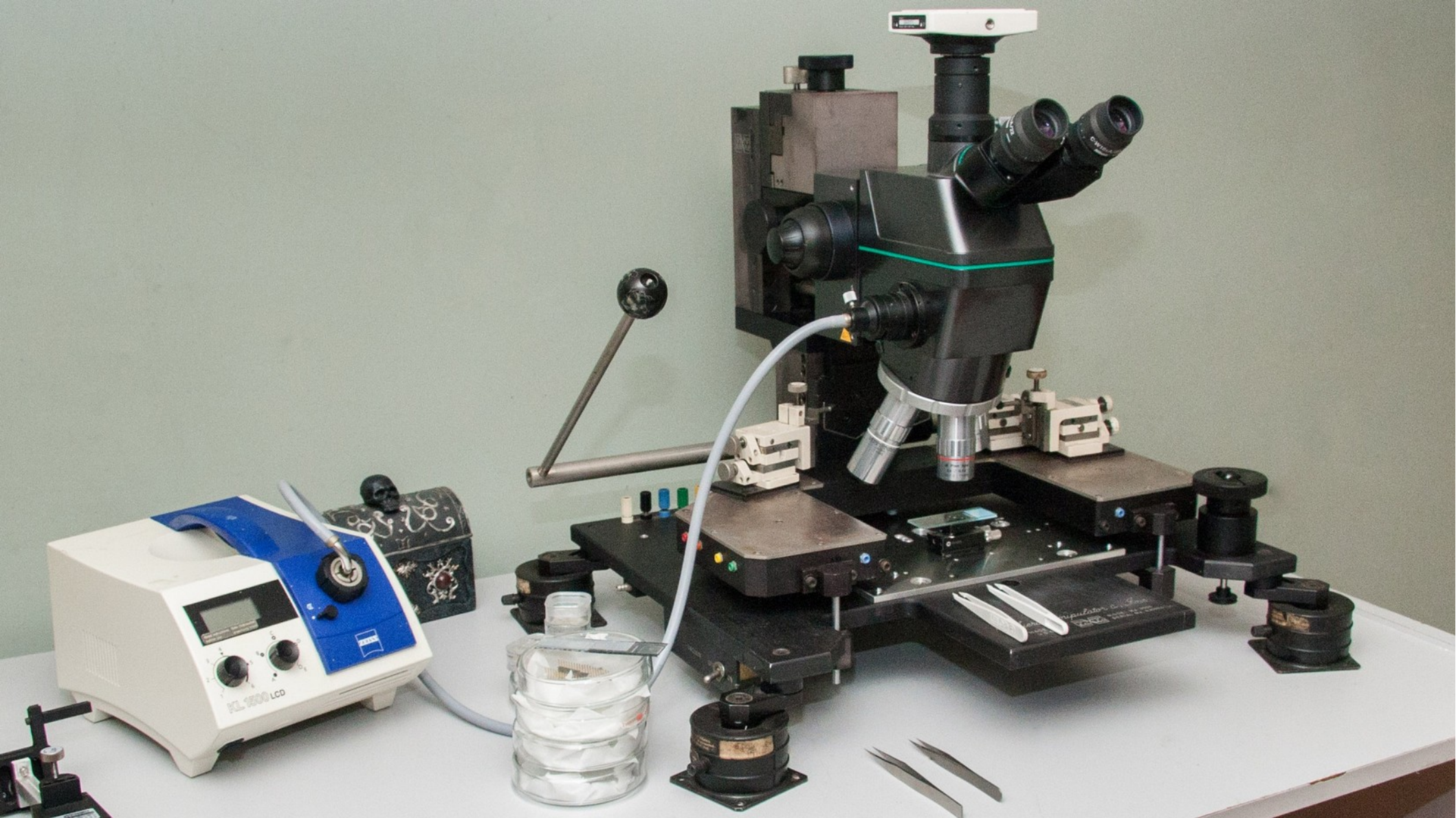
```
< 80 04 00 04
> 55 04 14 10 20 FC FD 00 00 00 00 00 00 00 00 40 00 00 00 00 00 00 61
      ^^^^^^ ^^ ^^
      ||      || Next category epoch
      ||      || Current category epoch
      Provider ID
                                         Checksum
```

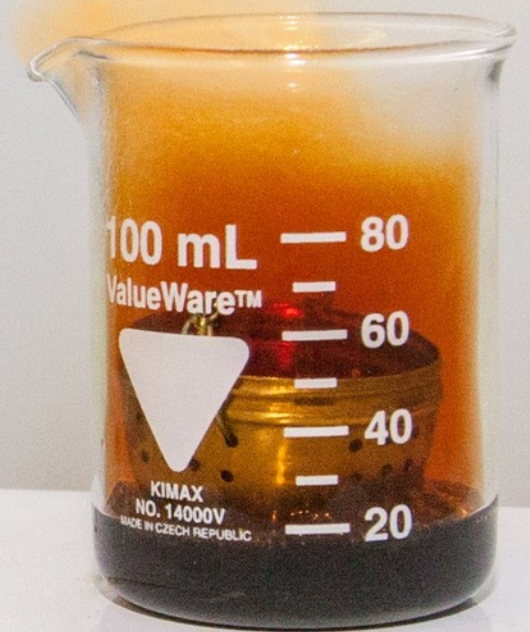
```
< 80 05 00 05
> 55 05 09 FE 16 00 10 D3 99 FA 20 00 64
      ^^^^^^^^^^^^^^^^^
      UA# - Unit Address of STB
```

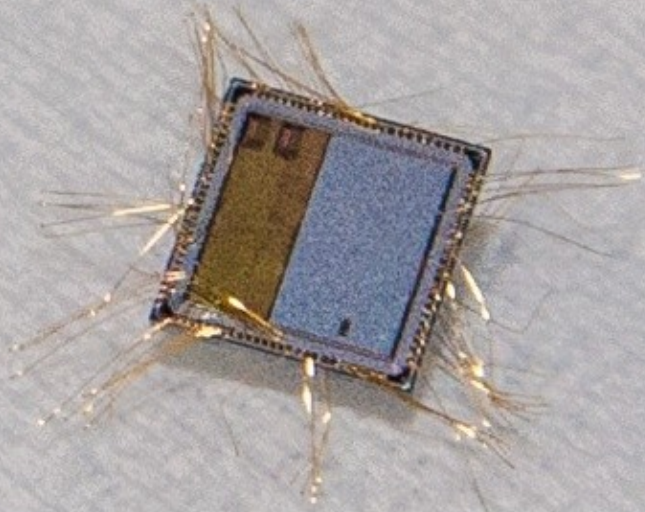
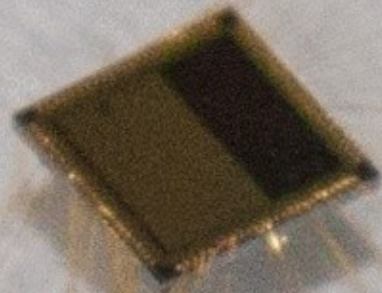
```
< 80 12 00 12
> 55 12 13 00 00 02 04 03 04 02 04 03 04 0C 6D 05 14 0D FC 00 00 00 80
      ^^^^^^^^^^^^^^ ^^^^^^^^^^^^^
      ||
      || Keyselect for next category epoch
      Keyselect for current category epoch
```

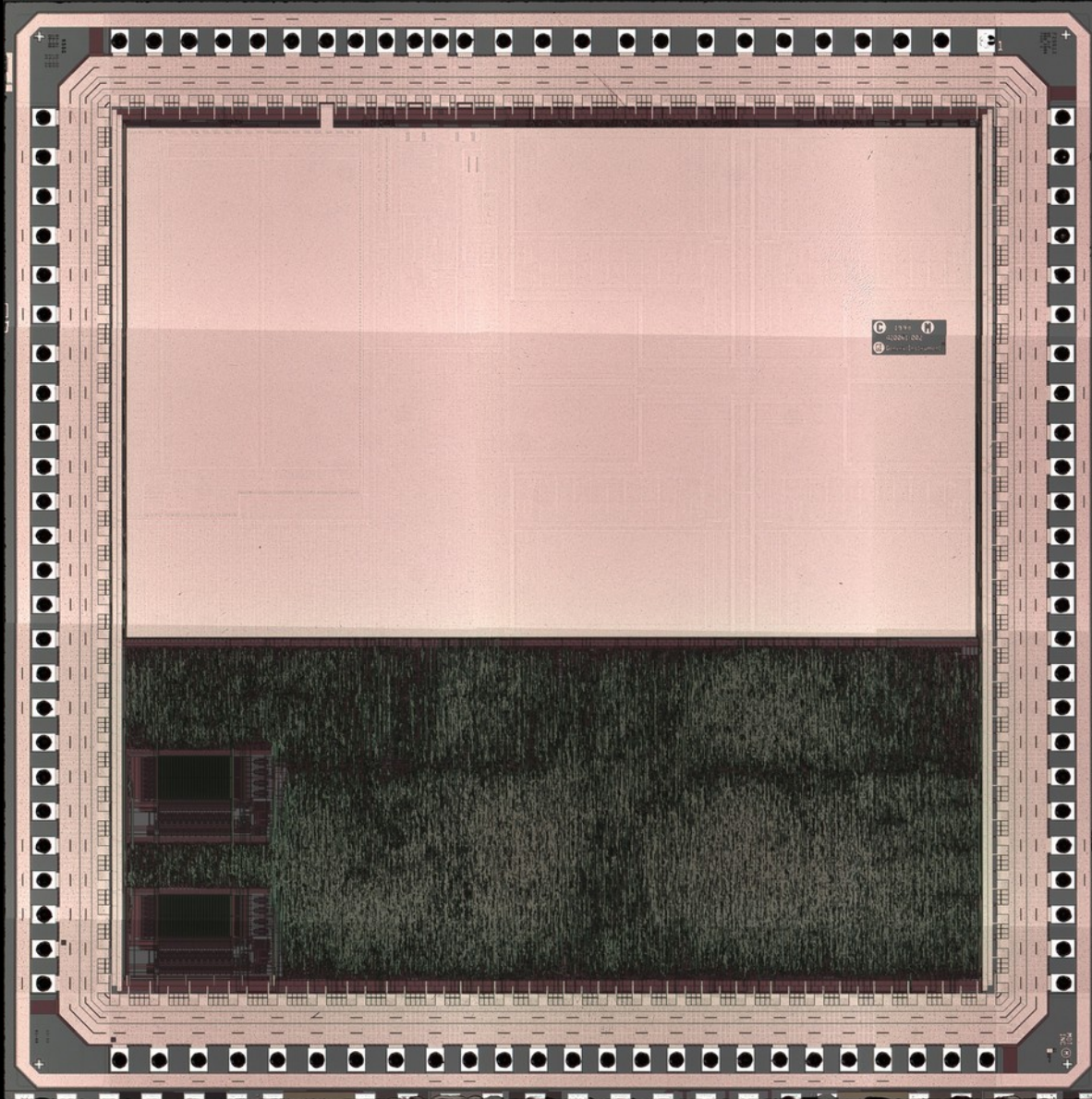
```
< 80 13 02 00 00 11
> 55 13 22 00 C0 4C 61 87 11 00 51 A1 00 DF 23 80 E0 01 98 D1 3F 00 80 41 00 08 20 48 00 00 00 D0 41 15 81 00 44 B1
      ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
      Subscription tiers
```

```
> 55 87 13 00 11 22 33 00 00 00 00 00 00 00 00 00 00 00
   ^^  ^^  ^^^^^^^^^^ ^^ ^^
   ||  ||  ||
   ||  ||  ||           ||  ||           Next program key epoch
   ||  ||  ||           ||  ||           Current program key epoch
   ||  ||  ||           Encryption mode - ZK / FP / FW S
   ||  ||  ||           Service ID of this channel
   ||  ||  ||           00 = Tuner0, 80 = Tuner1
07 = Response to Cmd07
87 = Unprompted response due to change in access rights (channel change etc)
```









whink[®]

RUST STAIN REMOVER

FOR ALL *white sinks*
COLORFAST *and white*
FABRICS *toilet bowls*

Not a general purpose rust remover.
Read precautions and directions before using.
Do not use this product if you do not intend to follow the directions.

DANGER: MAY BE FATAL OR CAUSE PERMANENT DAMAGE. CAUSES SEVERE BURNS WHICH MAY NOT BE IMMEDIATELY PAINFUL OR VISIBLE. VAPOR HARMFUL. CONTAINS HYDROFLUORIC ACID. KEEP OUT OF REACH OF CHILDREN. Use only with heavy duty household rubber gloves. Read back panel for additional precautions and directions.

10 fl oz (296 mL)



500 mL 9624-02

Nitric Acid, Fuming, 90%

Acide Nitrique, Fumant
BAKER ANALYZED[®] A.C.S. Reagent
HNO₃ FW 63.01
LOT H38N67

Meets Reagent Specifications for testing USP/NF Nitrofu-...

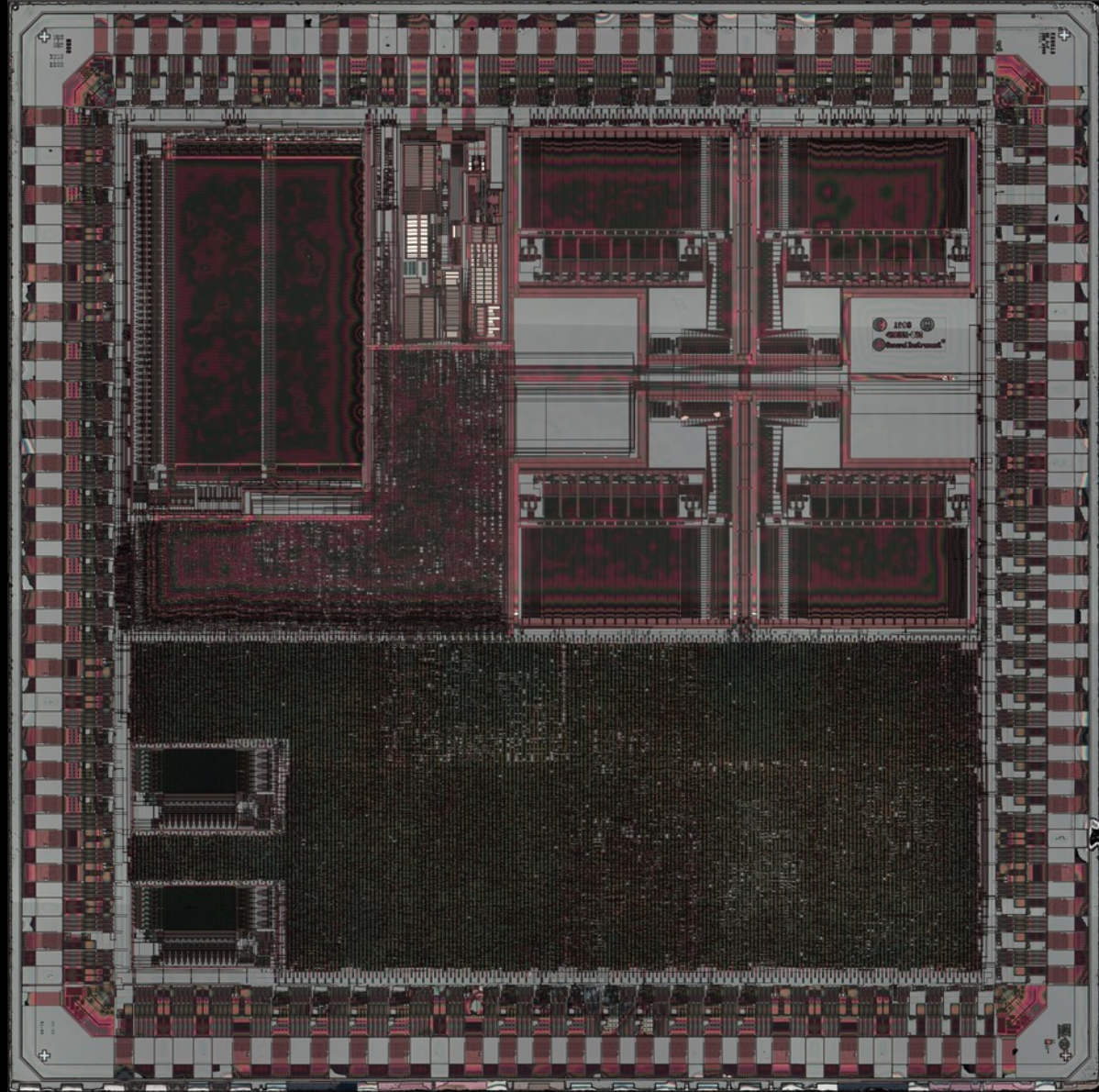
Meets A.C.S. Specifications	
Assay (as HNO ₃) (by acidimetry)	90.0 % min.
Distillation Test	Passes Test
Residue after ignition	0.002 % max.
Dissolved Oxides (as N ₂ O ₅)	0.1 % max.
Total Impurities (in ppm)	
Chloride (Cl)	0.7 max.
Sulfate (SO ₄)	5 max.
Arsenic (As)	0.3 max.
Heavy Metals (as Pb)	5 max.
Iron (Fe)	2 max.

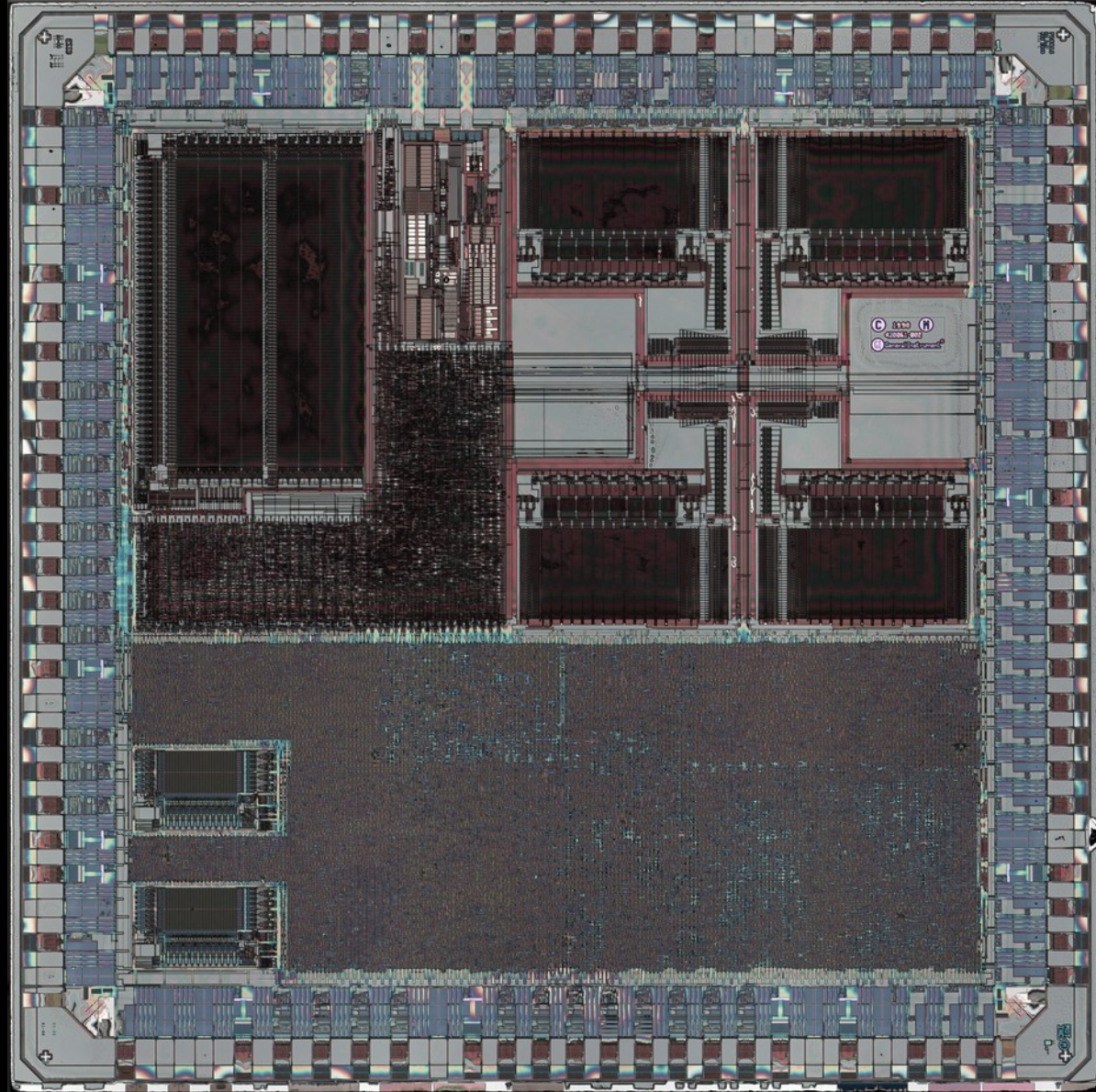
For Laboratory, Research or Manufacturing Use

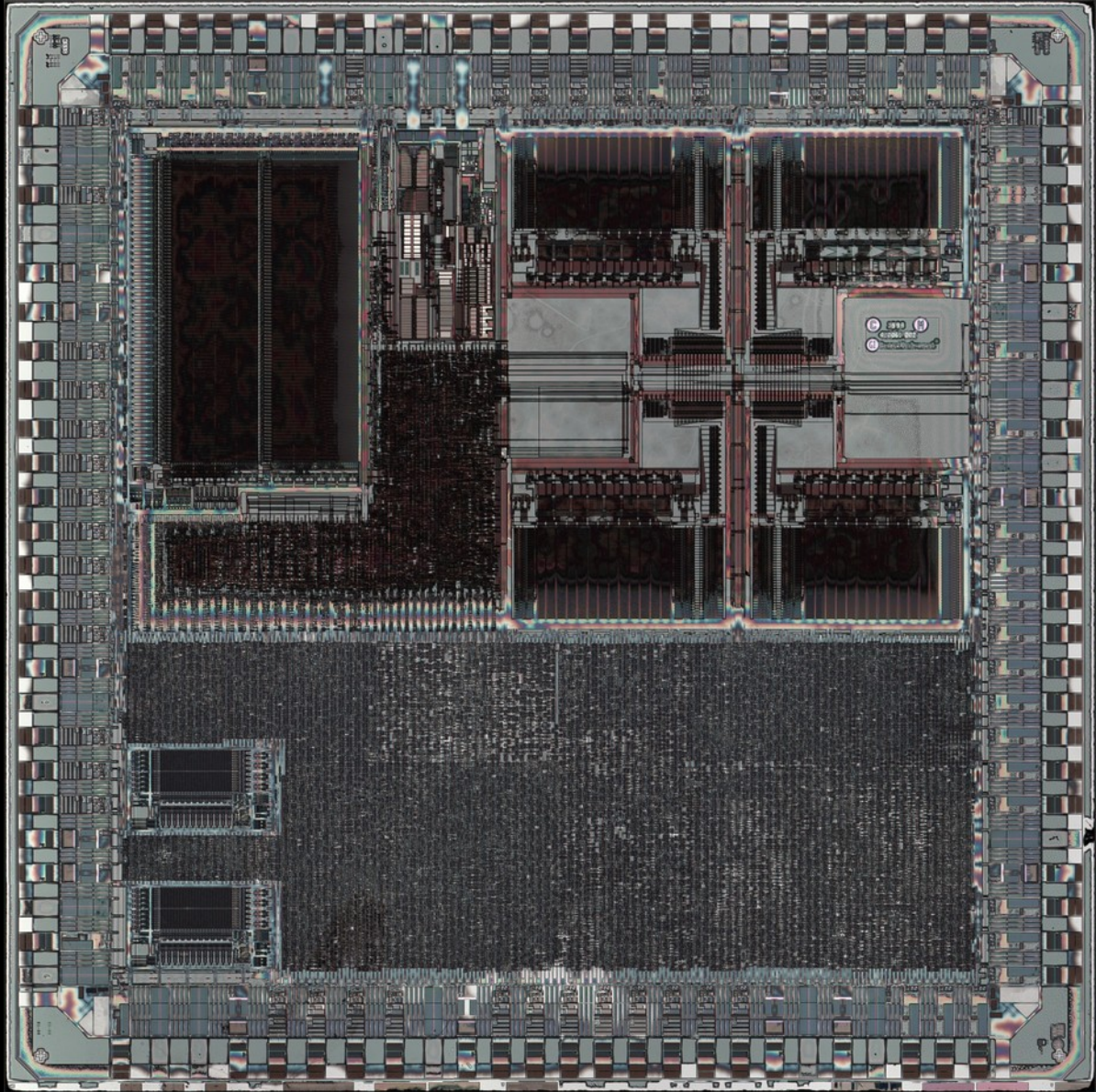


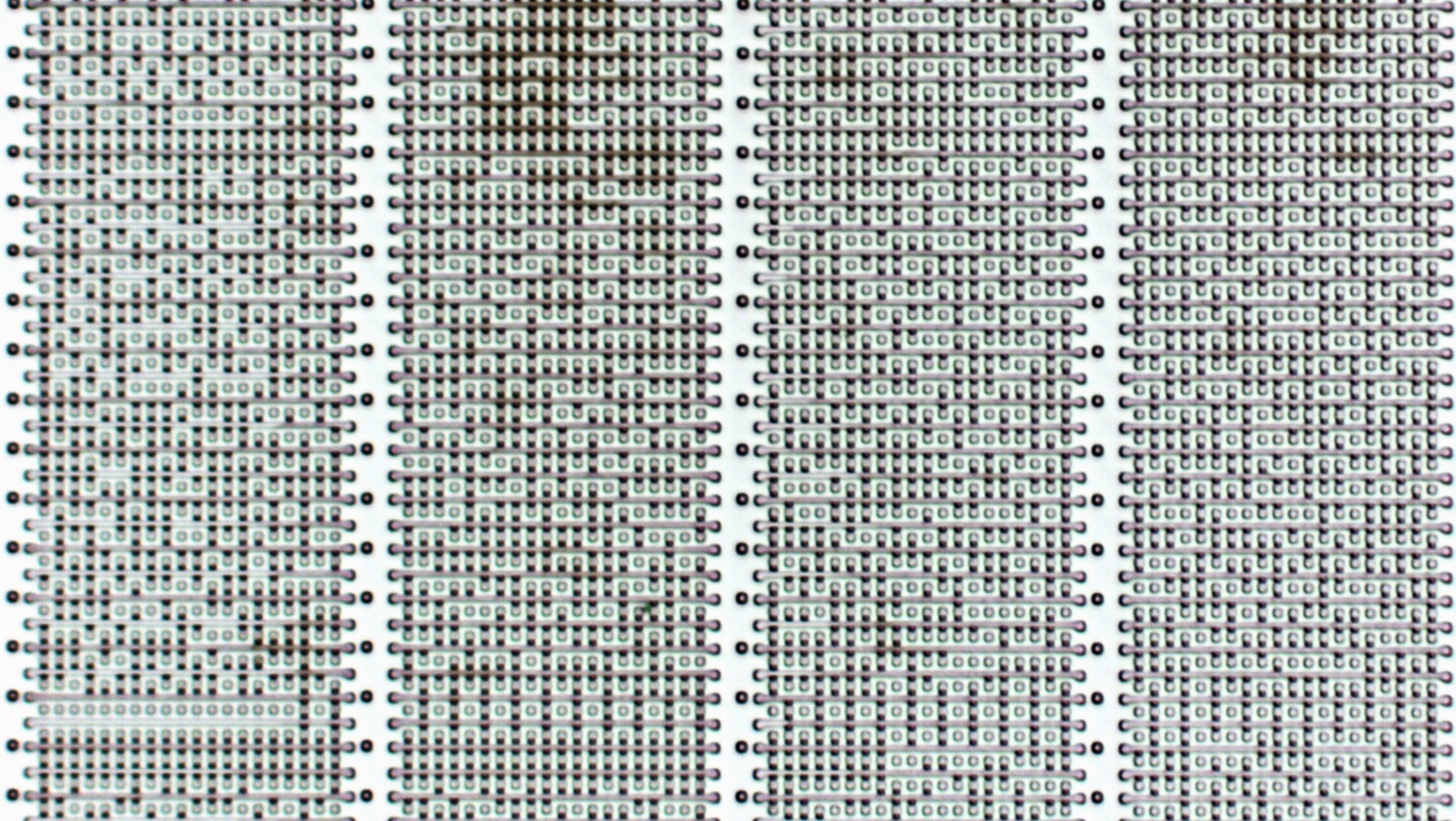
MADE IN USA
CAUTION: 7997-07-0

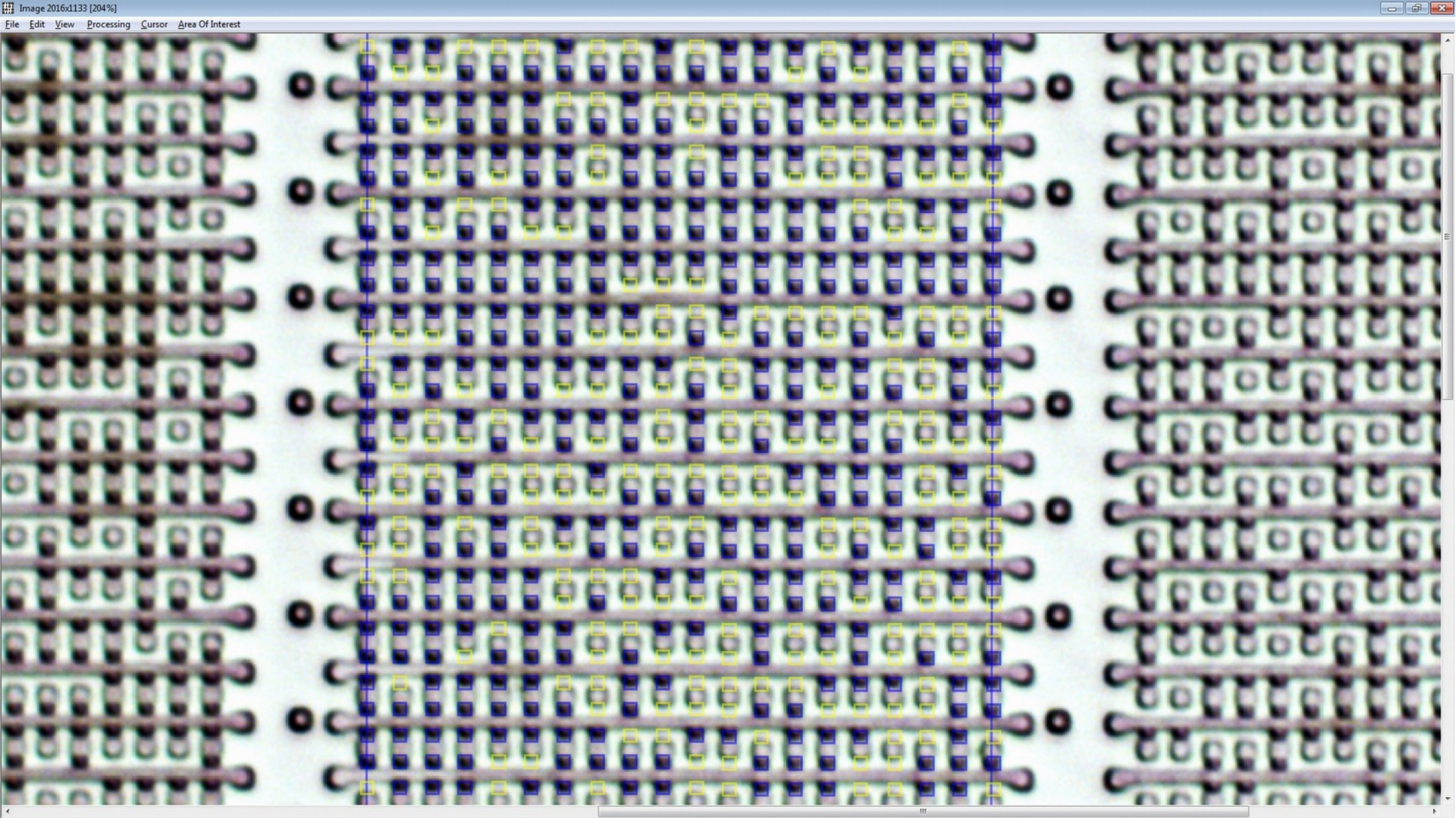
Philippine Chemicals, Inc.
Ph. 800-88-1111
www.baker.com



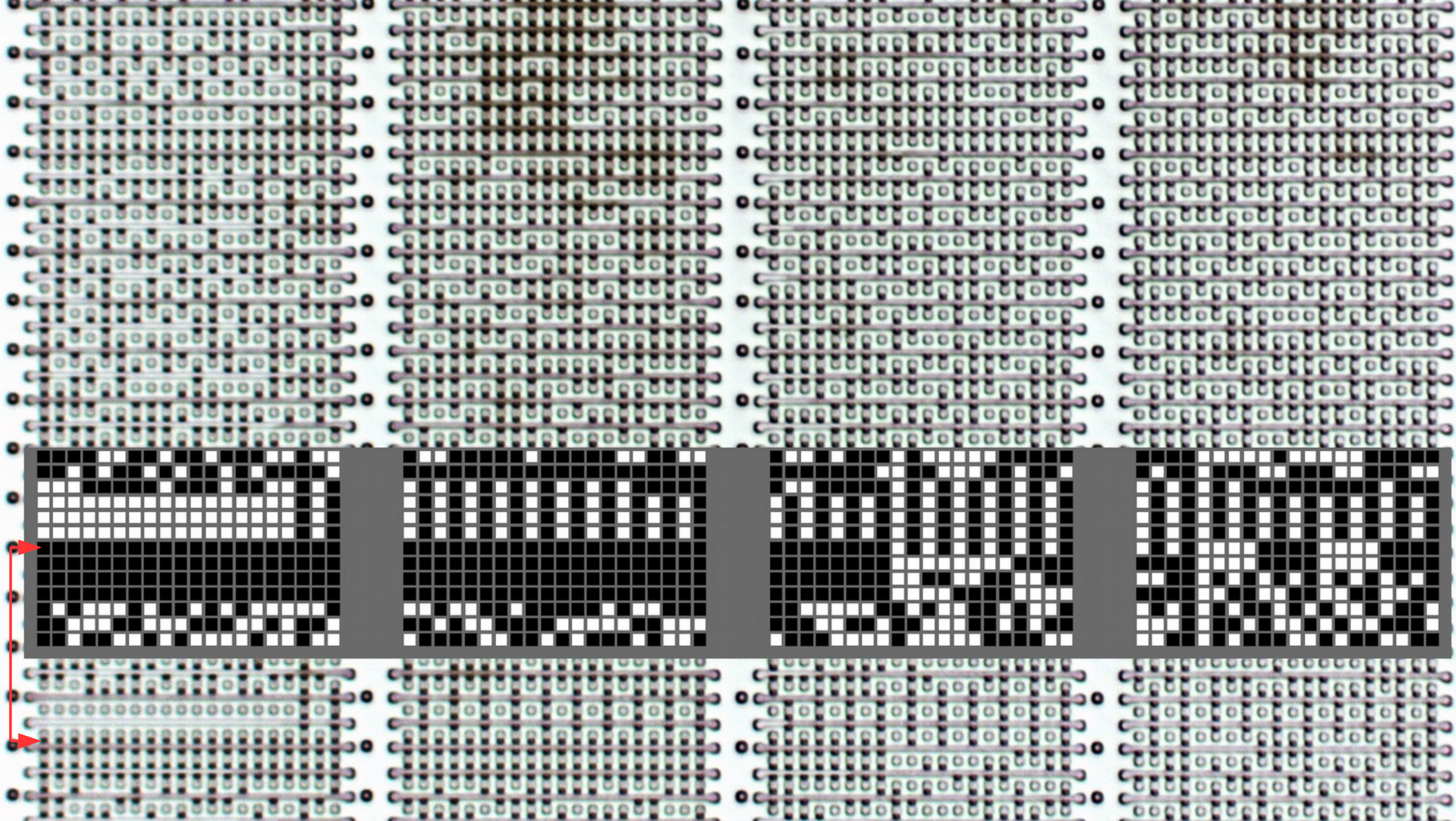


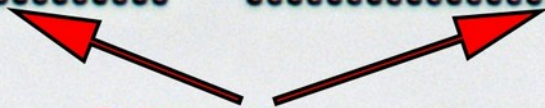
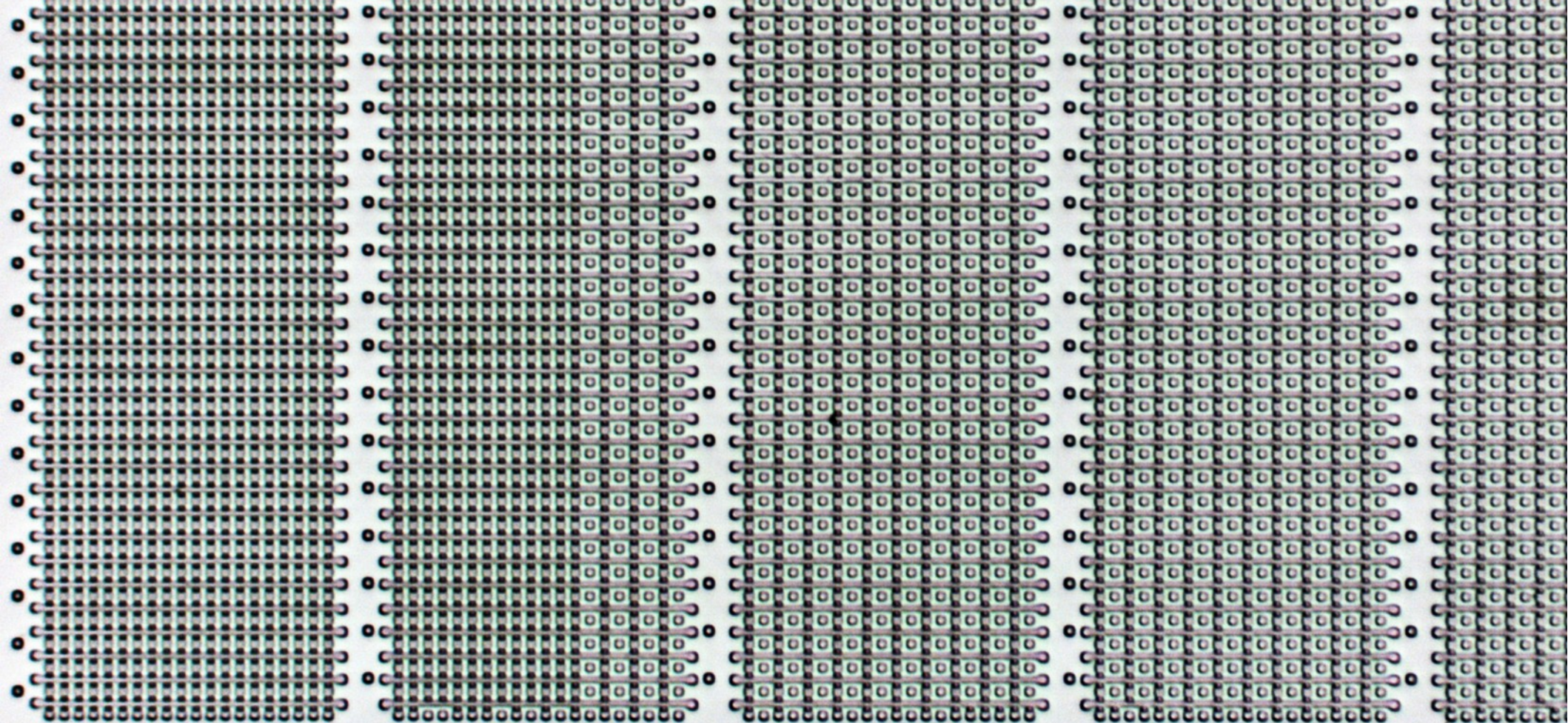












Vectors

Software disassembly (IDA)

The screenshot displays the IDA Pro disassembler interface for a file named 'payload.bin'. The main window shows assembly code for the 'SPI_RAM' section, with addresses ranging from 4000 to 4040. The code includes several subroutines and loops:

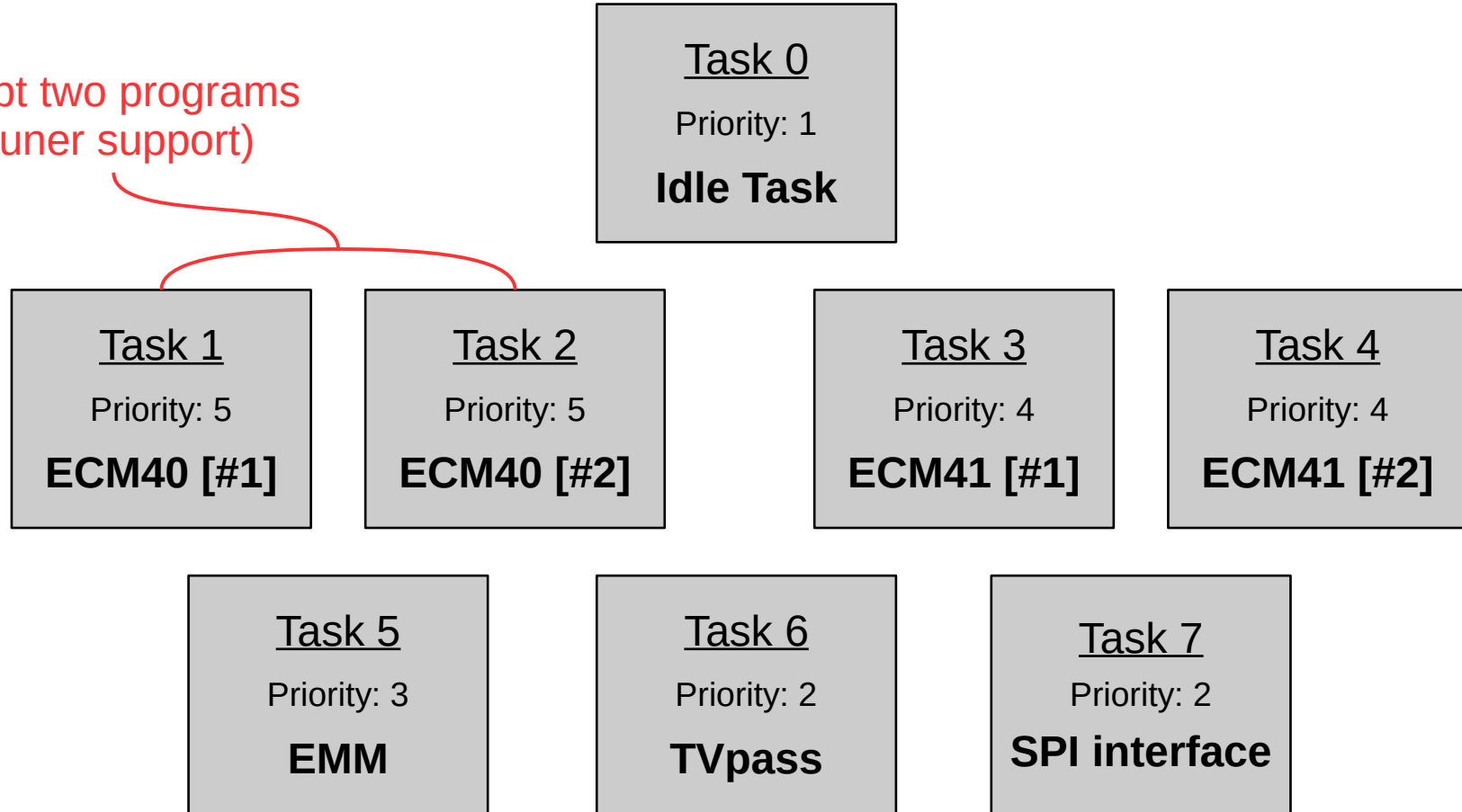
- executable_command:** A series of NOP instructions followed by a comment: "; pad the beginning of the packet with some NOPs to give a big of an entry point as if 'nop slide'".
- copy_intercept_loop:** A loop that copies code from SPI_RAM to main RAM. It uses instructions like LDA, STA, INX, and CFX.
- intercept_loop:** A loop that receives and processes packets. It includes instructions like JSR, RMB, LDX, PHX, LDA, BRK, and BRA.

On the right side, a hex view window shows a memory dump consisting of a grid of 00 00 00 00 bytes. At the bottom, an output window displays the IDAPython console output, which includes the following error message:

```
Python
IDAPython 64-bit v1.7.0 final (serial U) (c) The IDAPython Team <idapython@googlegroups.com>
Python
NO: idle Down Disk: 241GB
UNKNOWN 000006A0: RAM:06A0
Database for file 'full.bin' is loaded.
Compiling file 'C:\IDAS\0\idc\ida.idc'...
Executing function 'main'...
Command "JumpAsk" failed
AU: idle Down Disk: 2041GB
```


Task switcher

Decrypt two programs
(dual tuner support)



Hardware Peripherals

SPI slave

- RX RAM buffer
- TX RAM buffer
- Flag set when transfer complete

Transport Stream PID filter

- SPI selects ECM PID
 - Tables placed in RAM and RX flag set
- SPI selects PIDs to decrypt
- Software selects Provider ID filter for ECM

DES hardware

- Hardware encrypt/decrypt
- 56 bit key, 64 bit data
- Standard and custom modes

TS descrambler

- Software sets working key
- Selected PIDs are decrypted
- DES standard and custom modes

System analysis

- ✓ MPEG Transport stream – ECM & EMM logging
- ✓ SPI bus – logging and understanding of ACP messages
- ✓ ROM dump (entire ACP firmware)
- ✓ Software disassembly and simulation
- ✓ Keys: Fixed keys found in ROM

Possible bit errors from optical extraction ruled out by valid checksum on key area

We are here

→ Understanding of ECM and EMM algorithms

✗ Keys: Seed keys, category keys, program keys only exist in RAM

ECM40 – Working Key

Three sequential ECM40 messages (133 ms apart)

```
40 00 11 00 2D 80 07 2B 16 FA 22 04 00 00 00 BB
40 00 11 00 2D 80 07 2B 16 FB 22 04 00 00 01 42
40 00 11 00 2D 80 07 2B 16 FC 22 04 00 00 00 D2
      ^^^^      ^^^^      ^^      ^^
      Service ID Frame Count HW Crypt Mode
```

Frame Count is used to generate Working Key

ECM41 – Program Key

41 40 41 10 20 00 2D 80 07 75 6E 2A CE 13 09 E3
40 9E F1 9F E7 76 9A 7E BC 00 00 00 08 05 00 01
35 03 00 00 06 03 00 00 83 03 00 00 A9 03 00 00
C0 03 00 02 54 03 01 00 AB 69 1C D1 12 A8 CE D5

1020 Provider ID

8-byte Initialization Vector

2D8007 Service ID

00 Hardware crypto select

75 Category Epoch

08 Bitmapped (bit 6 indicates PPV)

6E Program Epoch

05 Number of acceptable tiers
(6, each 4 bytes)

2ACE13 Validity start

09E340 Valid period (24 hr.)

8-byte Encrypted Program Key

40 00 11 00 2D 80 07 2B 16 FA 22 04 00 00 00 BB

ECM40 Frame Count 2B16FA

41 40 41 10 20 00 2D 80 07 75 6F 34 B1 53 09 E3

ECM41 Validity Start 34B153 (next key)

40 44 37 29 A0 2E 25 4B A2 00 00 00 08 05 00 01

35 03 00 00 06 03 00 00 83 03 00 00 A9 03 00 00

C0 03 00 02 54 03 01 00 E8 F1 E4 74 45 B9 85 73

40 00 11 00 2D 80 07 2B 16 FB 22 04 00 00 01 42

ECM40 Frame Count 2B16FB

41 40 41 10 20 00 2D 80 07 75 6E 2A CE 13 09 E3

ECM41 Validity Start 2ACE13 (current key)

40 9E F1 9F E7 76 9A 7E BC 00 00 00 08 05 00 01

35 03 00 00 06 03 00 00 83 03 00 00 A9 03 00 00

C0 03 00 02 54 03 01 00 AB 69 1C D1 12 A8 CE D5

41 40 41 10 20 00 2D 80 07 75 6F 34 B1 53 09 E3

ECM41 Validity Start 34B153 (next key)

40 44 37 29 A0 2E 25 4B A2 00 00 00 08 05 00 01

35 03 00 00 06 03 00 00 83 03 00 00 A9 03 00 00

C0 03 00 02 54 03 01 00 E8 F1 E4 74 45 B9 85 73

40 00 11 00 2D 80 07 2B 16 FC 22 04 00 00 00 D2

ECM40 Frame Count 2B16FC

41 40 41 10 20 00 2D 80 07 75 6E 2A CE 13 09 E3

ECM41 Validity Start 2ACE13 (current key)

40 9E F1 9F E7 76 9A 7E BC 00 00 00 08 05 00 01

35 03 00 00 06 03 00 00 83 03 00 00 A9 03 00 00

C0 03 00 02 54 03 01 00 AB 69 1C D1 12 A8 CE D5

40 00 11 00 2D 80 07 2B 16 FD 22 04 00 00 01 58

ECM40 Frame Count 2B16FD

EMM95 – delivered in 4 parts

0x0013274945 = UA: 0000 3213 41765

```
95 10 A6 00 13 27 49 45 00 00 00 00 00 00 BE C0
00 75 00 00 40 29 20 8E 2D 88 B1 6B 44 1C 10 C3
C3 FB 21 B3 EB 02 04 03 04 02 04 03 04 0D 6C 09
90 0D AA 00 00 00 00 00 00 00 00 00 00 02 10 20
00 80 01 DF 00 60 87 03 00 50 F1 00 DF 23 00 80
00 80 91 33 80 00 40 00 00 22 48 00 00 01 00 41
05 80 00 44 0F 00 FF FF FF 01 FF FF FF 02 FF FF
FF 03 FF FF FF 04 FF FF FF 05 FF FF FF 06 FF FF
FF 07 FF FF FF 08 FF FF FF 09 FF FF FF 0A FF FF
FF 0B FF FF FF 0C FF FF FF 0D FF FF FF 0E FF FF
FF 0F FF FF FF
```

```
95 10 94 00 13 27 49 45 00 00 00 00 00 02 84 C0
00 75 00 00 40 BC 98 1E 16 6E 0E A0 1A 67 99 71
F2 8B 53 5E 25 02 04 03 04 02 04 03 04 02 00 03
DF 00 00 04 00 00 00 00 00 10 00 02 40 42 00 00
80 40 00 02 1A C0 1F 00 A0 07 B8 7B D4 0C 80 FF
87 DF E0 3F 00 E0 07 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 DF 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00
```

```
95 10 93 00 13 27 49 45 00 00 00 00 00 01 86 C0
00 75 00 00 40 75 77 07 CD E5 79 84 A7 12 BD 46
30 60 41 D2 B2 02 04 03 04 02 04 03 04 01 00 01
DF 30 14 00 00 00 00 C0 02 00 00 00 81 13 5E 20
1C 42 00 00 46 20 00 00 00 10 40 00 00 00 08 80
2F 0F 10 FF FF FF 11 FF FF FF 12 FF FF FF 13 FF
FF FF 14 FF FF FF 15 FF FF FF 16 FF FF FF 17 FF
FF FF 18 FF FF FF 19 FF FF FF 1A FF FF FF 1B FF
FF FF 1C FF FF FF 1D FF FF FF 1E FF FF FF 1F FF
FF FF
```

```
95 10 94 00 13 27 49 45 00 00 00 00 00 03 84 C0
00 75 00 00 40 DB 9B 83 88 DA 80 81 FE CD 14 50
FC A7 ED 27 69 02 04 03 04 02 04 03 04 05 40 03
DF 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 DF 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 DF 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00
```

```

95 10 A6 00 13 27 49 45 00 00 00 00 00 00 BE C0
00 75 00 00 40 29 20 8E 2D 88 B1 6B 44 1C 10 C3
C3 FB 21 B3 EB 02 04 03 04 02 04 03 04 0D 6C 09
90 0D AA 00 00 00 00 00 00 00 00 00 00 02 10 20
00 80 01 DF 00 60 87 03 00 50 F1 00 DF 23 00 80
00 80 91 33 80 00 40 00 00 22 48 00 00 01 00 41
05 80 00 44 0F 00 FF FF FF 01 FF FF FF 02 FF FF
FF 03 FF FF FF 04 FF FF FF 05 FF FF FF 06 FF FF
FF 07 FF FF FF 08 FF FF FF 09 FF FF FF 0A FF FF
FF 0B FF FF FF 0C FF FF FF 0D FF FF FF 0E FF FF
FF 0F FF FF FF

```

0013274945 Unit Address (UA)

00 EMM part #

BEC0 Bitmap indicating data items to follow

00 Hardware crypto select

75 Category Epoch

8-byte Encrypted Category key

8-byte Encrypted Category key (next)

02040304 Keyselect

02040304 Keyselect (next)

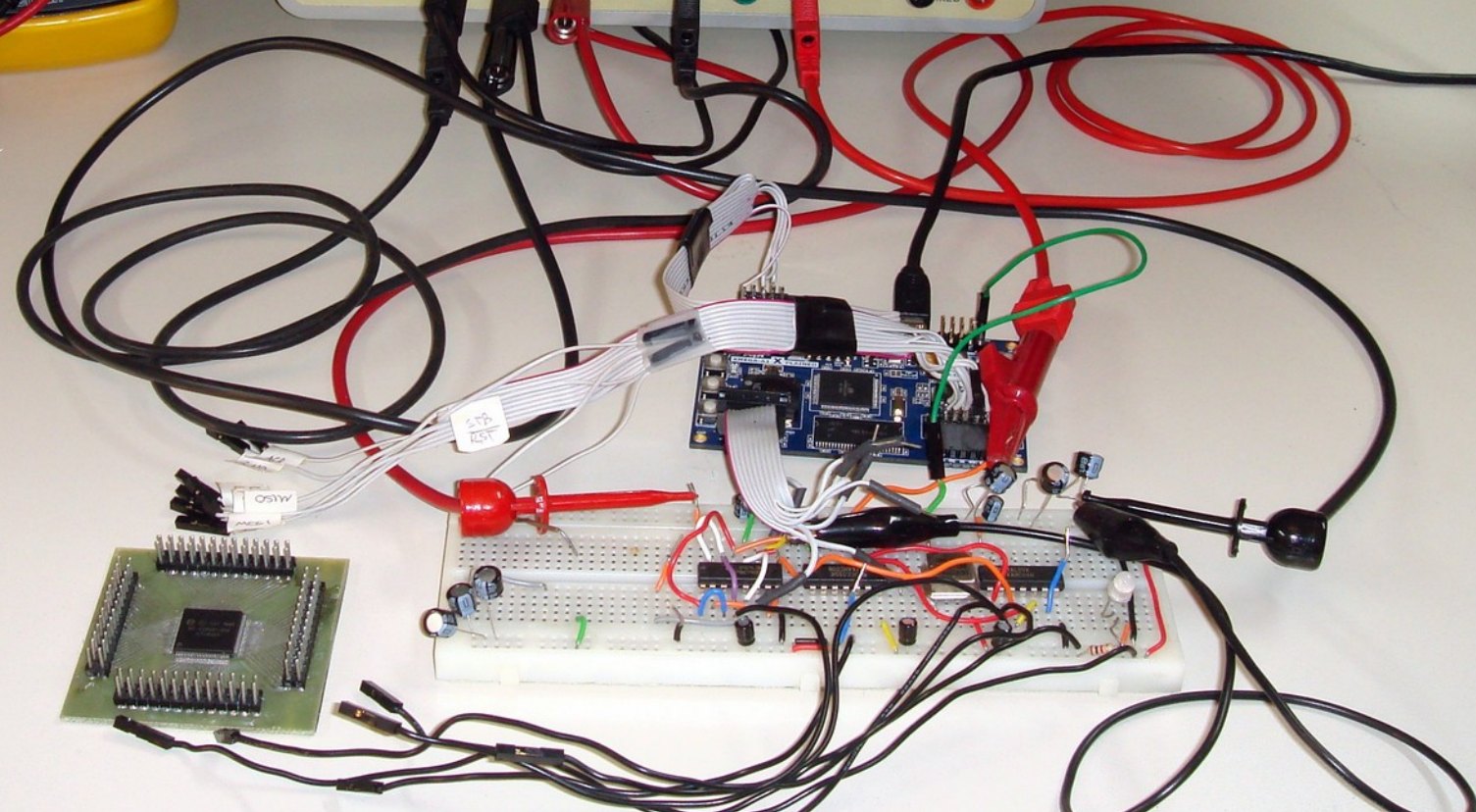
7-byte Geographic location

1020 Provider ID

00 Start of tier bank 0

Decrypting fixed-key channels

- Fixed-key channels can be decrypted with a STB if it was authorized in the past – it does not need current authorization (category key is not required)
- Fixed Program and Fixed Working keys are known (ROM)
 - Fixed keys verified by ROM checksum
- I didn't have access to any signals without hardware customization enabled for the DES descrambler



Normal	SPI RX:	[7289]:	55	05	09	FC	16	C2	88	48	74	58	85	84	C9	
Glitch	4/11412	[7293]:	55	13	09	FC	16	C2	88	48	74	58	85	84	DF	
Glitch	4/11418	[7292]:	55	FF	09	FC	16	C2	88	48	74	58	85	84	33	
Glitch	4/11419	[7290]:	55	BF	09	FC	16	C2	88	48	74	58	85	84	73	
Glitch	4/11436	[7435]:	55	13	22	00	00	00	00	00	00	00	00	00	00	00
Glitch	4/11439	[7292]:	55	13	09	FC	16	C2	88	48	74	58	85	84	DF	
Glitch	4/11452	[7442]:	55	05	22	FC	16	C2	88	48	74	58	85	84	00	00 00 00 00 00 00 00 00 00 00
Glitch	4/11456	[7437]:	55	05	22	FC	16	C2	88	48	74	58	85	84	00	00 00 00 00 00 00 00 00 00 00
Glitch	4/11483	[11636]:	55	05	09	00	00	00	00	00	00	00	00	00	0C	
Glitch	4/11491	[7286]:	55	05	09	00	16	C2	88	48	74	58	85	84	35	
Glitch	4/11506	[7285]:	55	05	09	FC	00	C2	88	48	74	58	85	84	DF	
Glitch	4/11514	[7293]:	55	05	09	FC	3E	C2	88	48	74	58	85	84	E1	
Glitch	4/11522	[7295]:	55	05	09	FC	00	C2	88	48	74	58	85	84	DF	
Glitch	4/11527	[7294]:	55	05	09	FC	BE	C2	88	48	74	58	85	84	61	
Glitch	4/11537	[7293]:	55	05	09	FC	00	C2	88	48	74	58	85	84	DF	
Glitch	4/11560	[7293]:	55	05	09	FC	16	C2	88	48	74	16	85	84	87	
Glitch	4/11578	[7292]:	55	05	09	FC	16	C2	88	48	74	00	85	84	91	
Glitch	4/11599	[7260]:	55	05	09	FC	16	00	00	00	00	58	00	78	C6	
Glitch	4/11622	[7291]:	55	05	09	FC	16	C2	88	48	58	58	85	84	E5	
Glitch	4/11626	[7286]:	55	05	09	FC	16	C2	88	48	21	58	85	84	9C	
Glitch	4/11654	[7292]:	55	05	09	FC	16	C2	88	48	00	58	85	84	BD	
Glitch	4/11677	[7271]:	55	05	09	FC	16	00	00	00	74	58	00	7C	B6	
Glitch	4/11685	[7292]:	55	05	09	FC	16	C2	88	74	74	58	85	84	F5	
Glitch	4/11694	[7286]:	55	05	09	FC	16	C2	88	00	74	58	85	84	81	
Glitch	4/11700	[7295]:	55	05	09	FC	16	C2	88	80	74	58	85	84	01	
Glitch	4/11741	[7277]:	55	05	09	FC	16	00	00	48	74	58	85	84	83	
Glitch	4/11746	[10738]:	55	85	09	FC	16	00	00	48	74	58	00	00	02	
Glitch	4/11751	[10740]:	55	05	09	FC	16	00	00	48	74	58	00	00	82	
Glitch	4/11767	[7292]:	55	05	09	FC	16	C2	FE	48	74	58	85	84	BF	
Glitch	4/11779	[7278]:	55	05	09	FC	16	00	88	48	74	58	85	84	0B	Checksum
Glitch	4/11790	[7293]:	55	05	09	FC	16	C2	00	48	74	58	85	84	41	
Glitch	4/11812	[10747]:	55	85	09	FC	16	00	88	48	74	58	00	00	8A	
Glitch	4/11843	[7289]:	55	05	09	FC	16	00	88	48	74	58	85	84	0B	
Glitch	4/11901	[7288]:	55	05	09	FC	16	C2	88	48	74	58	00	84	4C	
Glitch	4/11937	[7290]:	55	05	09	FC	16	C2	88	48	74	58	85	00	4D	

Normal (no glitch) response

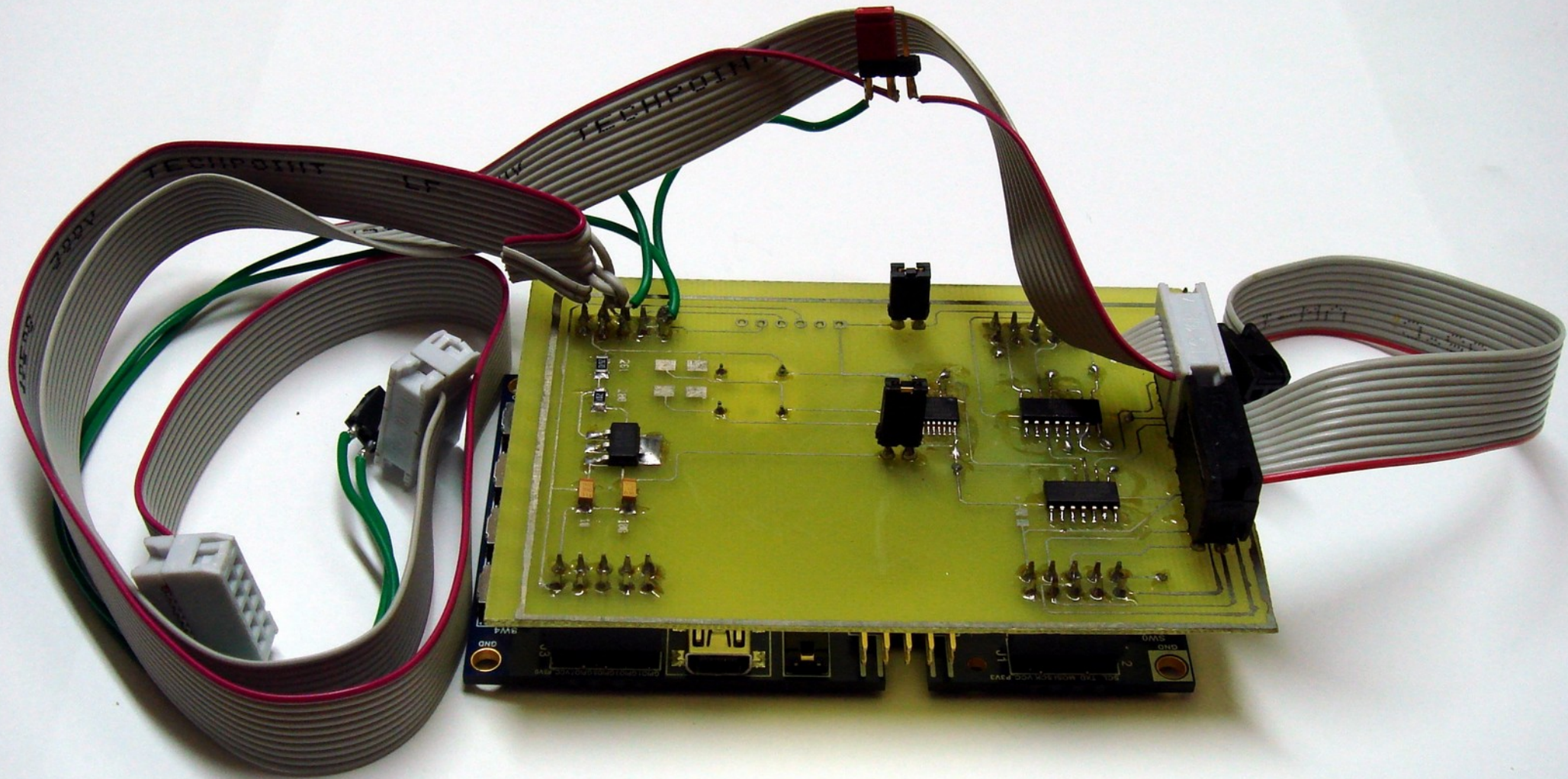
Responses changed by glitch

Checksum

Time until READY

Delay before glitch





Glitch payload - RAM intercept loop

(addresses have been changed)

```
SPI_RAM:4000 05          .BYTE 5          ; cmd05
SPI_RAM:4001 40          .BYTE $40         ; length 0x40 bytes
SPI_RAM:4002          ; -----
SPI_RAM:4002          executable_command:      ; pad the beginning of the packet with some NOPs to give as big of an entry point as possible
SPI_RAM:4002 EA          NOP
SPI_RAM:4003 EA          NOP          ; "nop slide"
SPI_RAM:4004 EA          NOP
SPI_RAM:4005 EA          NOP
SPI_RAM:4006 EA          NOP
SPI_RAM:4007 EA          NOP
SPI_RAM:4008 EA          NOP
SPI_RAM:4009 EA          NOP
SPI_RAM:400A EA          NOP
SPI_RAM:400B EA          NOP
SPI_RAM:400C EA          NOP
SPI_RAM:400D A2 00      LDX      #0
SPI_RAM:400F          copy_intercept_loop:      ; CODE XREF: SPI_RAM:4018
SPI_RAM:400F BD 2C 40    LDA      $402C,X      ; copy intercept loop from SPI_RAM to main RAM so it isn't overwritten by next packet
SPI_RAM:4012 9D 40 11    STA      $1140,X
SPI_RAM:4015 E8          INX
SPI_RAM:4016 E0 20      CPX      #$20 ; ' '
SPI_RAM:4018 D0 F5      BNE     copy_intercept_loop ; copy code from SPI_RAM to main RAM
SPI_RAM:401A A9 AB      LDA      #$AB ; ' '
SPI_RAM:401C 8D 14 41    STA      $4114      ; setup response at 3114: AB 00 AB
SPI_RAM:401F A9 00      LDA      #0
SPI_RAM:4021 8D 15 41    STA      $4115
SPI_RAM:4024 A9 AB      LDA      #$AB ; ' '
SPI_RAM:4026 8D 16 41    STA      $4116
SPI_RAM:4029 4C 40 11    JMP      $1140      ; jump to code that was copied from SPI_RAM to main RAM
SPI_RAM:402C          ; -----
SPI_RAM:402C          intercept_loop:
SPI_RAM:402C 20 24 A2    JSR      $A224      ; send packet we prepped in SPI_RAM (calling routine from the middle to skip over undesired behaviour)
SPI_RAM:402F 20 D2 A5    JSR      $A5D2      ; reset interrupt flag and clear interrupt mask
SPI_RAM:4032 17 E1      RMB1     $E1        ; clear bit indicating response has been sent (no further packet waiting to be processed)
SPI_RAM:4034 A2 00      LDX      #0
SPI_RAM:4036 DA          PHX
SPI_RAM:4037 A2 E1      LDX      #$E1      ; push 0x00 on stack
SPI_RAM:4039 DA          PHX          ; push address of flag for us to be awoken on (received SPI packet)
SPI_RAM:403A A9 02      LDA      #2
SPI_RAM:403C 00          BRK
SPI_RAM:403D          ; -----
SPI_RAM:403D 20 02 40    JSR      executable_command ; call subroutine received in RAM
SPI_RAM:4040 80 EA      BRA      intercept_loop ; loop to receive next executable command
```



Bit errors found in optical ROM dump

Result of ROM extraction from single sample

Errors marked in red

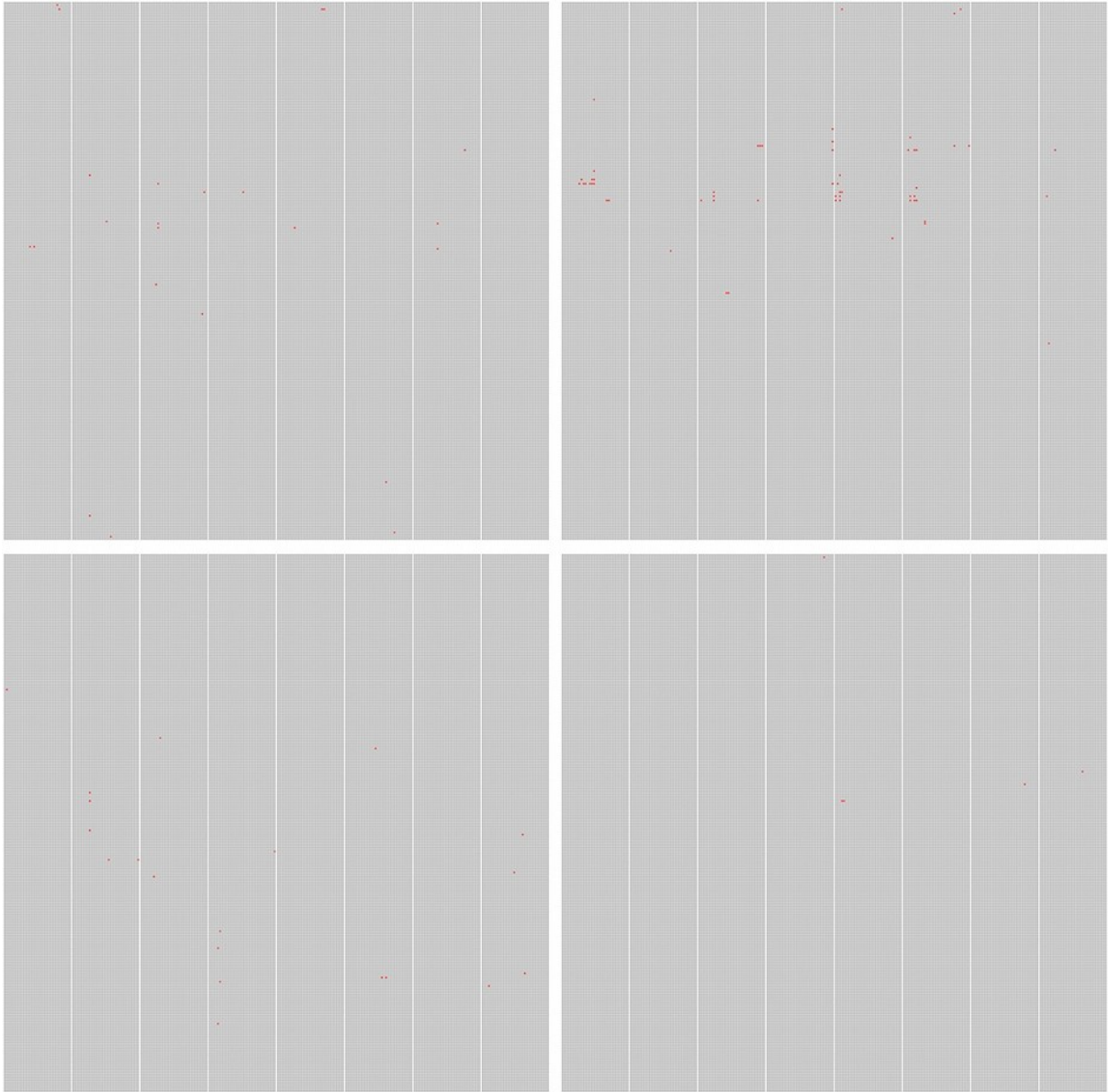
By the bit

262,144 bits total
105 bit errors
= 0.04 % error rate (99.96% accuracy)

By the byte

32,768 bytes total
104 byte errors
= 0.32 % error rate (99.68% accuracy)

Only one byte had more than a single flipped bit.



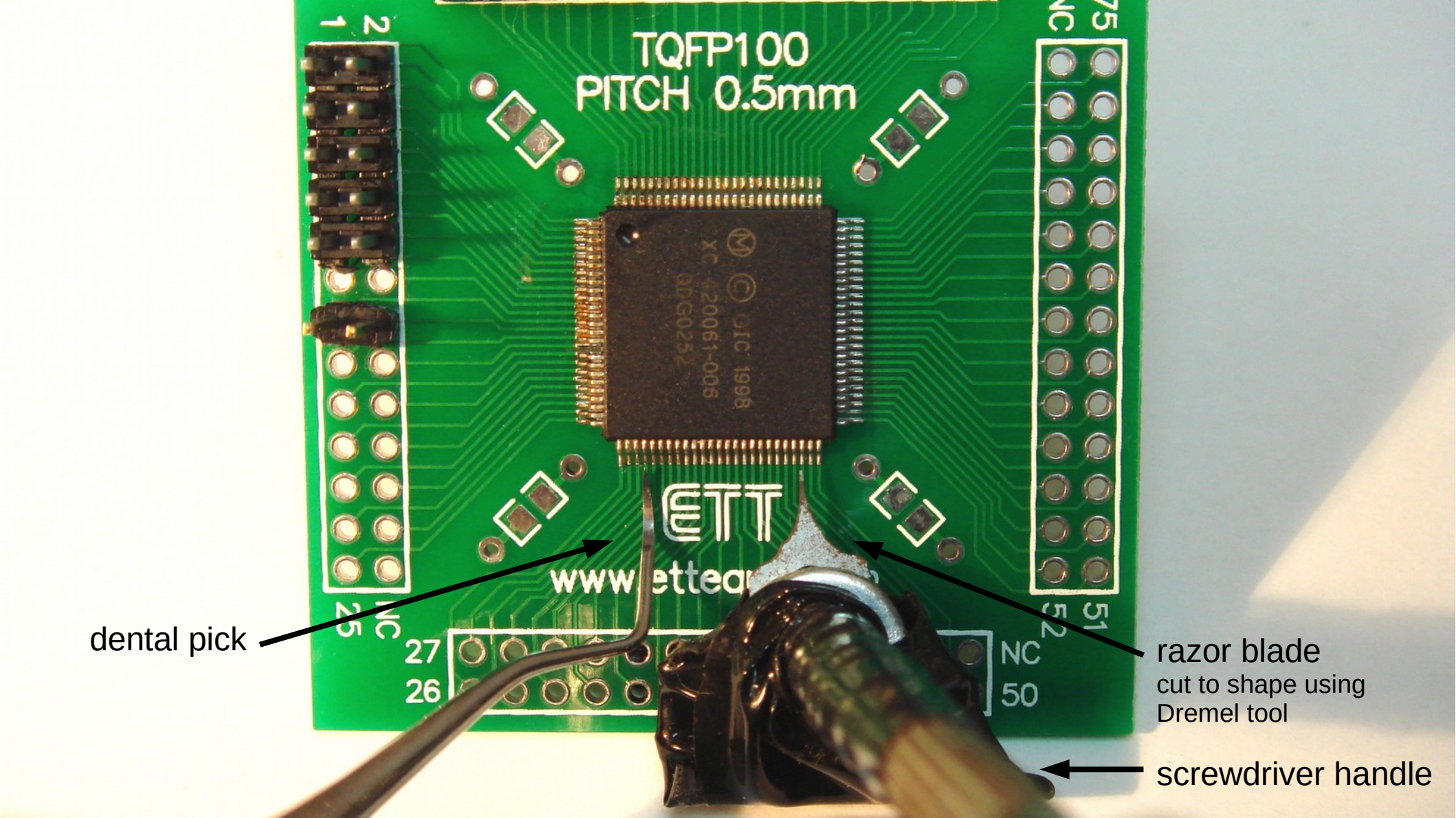
Key extraction from RAM

- All keys are stored only in RAM.
- RAM is battery-backed.
- Until now, glitching has been done on chips desoldered and put on test board.
 - This isolates the 16 VCC pins from each other and other components.
- RAM contents lost when chip is desoldered.
- If we can successfully glitch ACP with RAM intact then all keys can be read.
 - Category and Seed Keys are desirable.

Adapt glitcher to work on ACP in-circuit

or

Remove ACP from STB without losing RAM contents



TQFP100
PITCH 0.5mm

XC © GIC 1998
XC #20061-008
00G0252

ETT

www.ettear.com

dental pick

razor blade
cut to shape using
Dremel tool

screwdriver handle

1
2

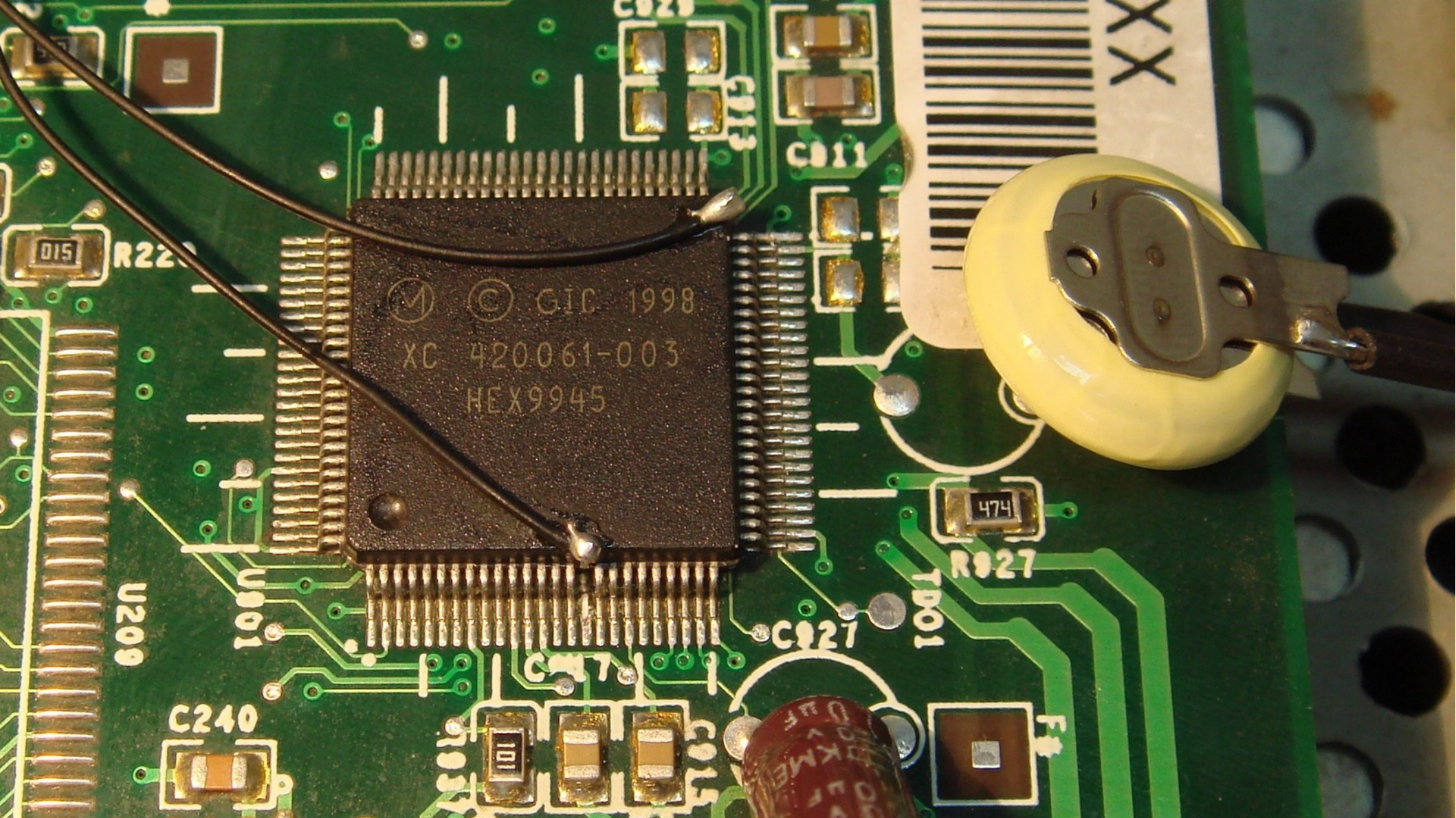
75
NC

NC
25

27
26

NC
50

51
52



© GIC 1998
XC 420061-003
HEX9945

XXX

474

R927

C927

T901

C240

101

C915

D15

R22

U209

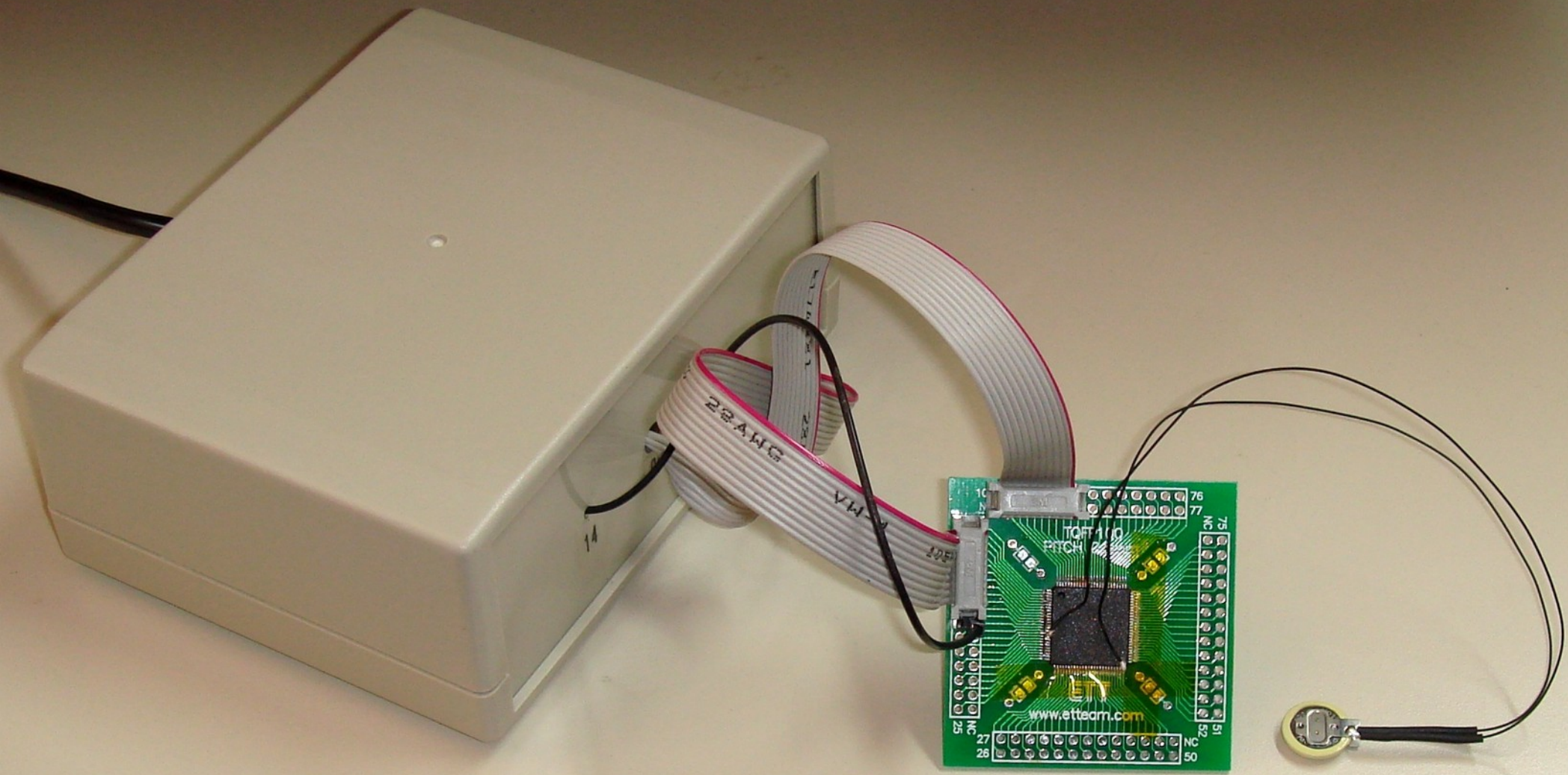
U901

C917

C929

C911

E



Transport Stream scrambling

188-byte TS packet

47	12	34	00	← 4-byte header																03	03	03	03	03	03	03	03	03	04	04	04	04	04	04	04	04
01	01	01	01	01	01	01	01	01	02	02	02	02	02	02	02	02	02	02	03	03	03	03	03	03	03	03	03	04	04	04	04	04	04	04	04	
05	05	05	05	05	05	05	05	05	06	06	06	06	06	06	06	06	06	06	07	07	07	07	07	07	07	07	07	08	08	08	08	08	08	08	08	
09	09	09	09	09	09	09	09	09	10	10	10	10	10	10	10	10	10	10	11	11	11	11	11	11	11	11	11	12	12	12	12	12	12	12	12	
13	13	13	13	13	13	13	13	13	14	14	14	14	14	14	14	14	14	14	15	15	15	15	15	15	15	15	15	16	16	16	16	16	16	16	16	
17	17	17	17	17	17	17	17	17	18	18	18	18	18	18	18	18	18	18	19	19	19	19	19	19	19	19	19	20	20	20	20	20	20	20	20	
21	21	21	21	21	21	21	21	21	22	22	22	22	22	22	22	22	22	22	23	23	23	23	23	23	23	23	23	23	23	23	23	23	23	23		

← 23 blocks of 8 bytes crypted data

Analyzing scrambling

- Flip bits in ciphertext to observe results in decrypted result (CBC/ECB/OFB modes)
 - *one bit flipped corrupts one 8-byte block plus corresponding bit in next block: CBC mode*
- Observe timing of decryption, look for changes in timing due to algorithm differences
 - *changes to algorithm such as number of rounds should have effect on timing*
- Use DES HW as oracle
 - *send test data through ACP, controlling data, key, and all H/W registers*
- Use DES weak keys (all 0 or all 1) and observe if behaviour matches standard DES
 - *encryption and decryption operations are equivalent when using key made of all 0 or all 1 bits*

Cracking Hardware Customization: DES key XOR taps

- Using H/W DES engine: Decrypt 23 blocks of 00 data, with 00 key, and customization enabled
- In software: Try decrypting 00 data with all combinations of key having 1 or more bits flipped
- Check for decryption matching any of the 23 blocks
- Positive results seen within seconds – some XOR masks have as few as 3 bits set

```
for( flip[0] = -1; flip[0] < 56; flip[0]++ ) { // loop through all bit positions
for( flip[1] = -1; flip[1]<56; flip[1]++ ) {
if( flip[1] <= flip[0] && flip[0] != -1 ) continue; // don't need to test with bits already tested flipped
for( flip[2] = -1; flip[2]<56; flip[2]++ ) {
if( flip[2] <= flip[1] && flip[1] != -1 ) continue;
for( flip[3] = -1; flip[3]<56; flip[3]++ ) {
if( flip[3] <= flip[2] && flip[2] != -1 ) continue;
for( flip[4] = -1; flip[4]<56; flip[4]++ ) {
if( flip[4] <= flip[3] && flip[3] != -1 ) continue;
for( flip[5] = -1; flip[5]<56; flip[5]++ ) {
if( flip[5] <= flip[4] && flip[4] != -1 ) continue;
for( flip[6] = -1; flip[6]<56; flip[6]++ ) {
if( flip[6] <= flip[5] && flip[5] != -1 ) continue;
for( flip[7] = -1; flip[7]<56; flip[7]++ ) {
if( flip[7] <= flip[6] && flip[6] != -1 ) continue;
for( flip[8] = -1; flip[8]<56; flip[8]++ ) {
if( flip[8] <= flip[7] && flip[7] != -1 ) continue;
for( flip[9] = -1; flip[9]<56; flip[9]++ ) {
if( flip[9] <= flip[8] && flip[8] != -1 ) continue;
for( flip[10] = -1; flip[10]<56; flip[10]++ ) {
if( flip[10] <= flip[9] && flip[9] != -1 ) continue;
for( flip[11] = -1; flip[11]<56; flip[11]++ ) {
if( flip[11] <= flip[10] && flip[10] != -1 ) continue;
for( flip[12] = -1; flip[12]<56; flip[12]++ ) {
if( flip[12] <= flip[11] && flip[11] != -1 ) continue;
for( flip[13] = -1; flip[13]<56; flip[13]++ ) {
if( flip[13] <= flip[12] && flip[12] != -1 ) continue;
for( flip[14] = -1; flip[14]<56; flip[14]++ ) {
if( flip[14] <= flip[13] && flip[13] != -1 ) continue;
for( flip[15] = -1; flip[15]<56; flip[15]++ ) {
if( flip[15] <= flip[14] && flip[14] != -1 ) continue;

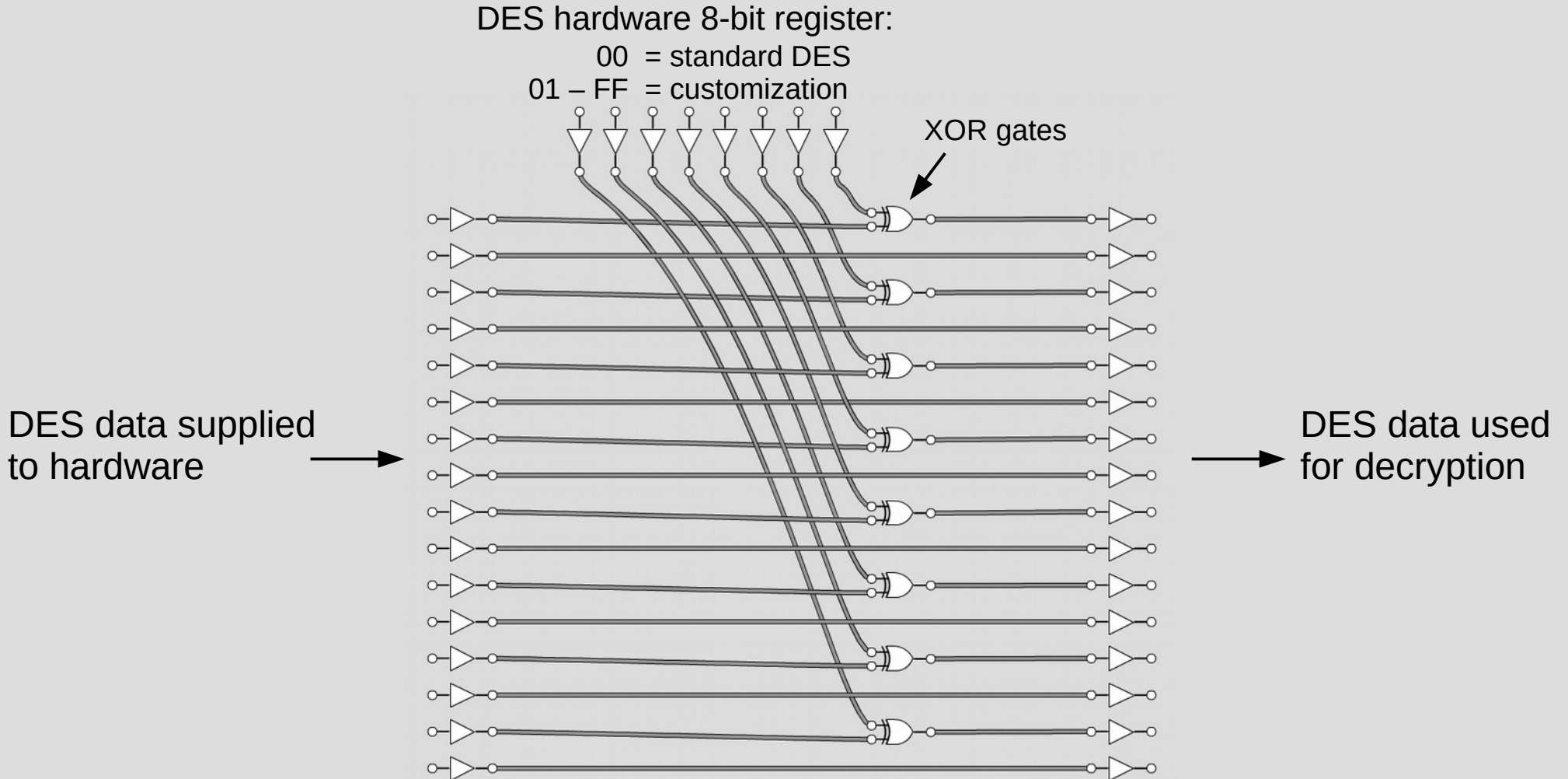
for( i=0; i<8; i++ ) {
des_key[i] = 0; // initialize des_key[] to all zero key
des_data[i] = 0; // initialize des_data[] to all zero ciphertext
}

for( i=0; i<16; i++ )
{
if( flip[i] >= 0 ) // XOR bits that are selected for this test
des_key[flip[i]/7] ^= 1<<((flip[i]%7)+1); // the +1 is because bit 0 is parity bit (unused)
}

ecb_decrypt( (char *)des_key, (char *)des_data, 8, DES_DECRYPT );

for( i=0; i<encrypted_blocks; i++ ) { // check des_data[] against encrypted blocks for a match
for( n=0; n<8; n++ )
```

Cracking Hardware Customization: DES data XOR taps



Softcam Implementation

- EMM processing
- ECM41 and ECM40 processing
- DES descrambler with hardware customization (key XOR)
- DES crypto peripheral with hardware customization (data XOR)
- Input – transport stream from satellite or cable interface
- Output – decrypted MPEG transport stream

VALID

-20dBFS
Source
16:9
1080i59

MCR'S THIS IS THE UFC 199 PAY-PER-VIEW PATH

CHANNEL 1 & 2 1KHZ TONE

CHANNEL 3 & 4 400 HZ TONE

Weaknesses

- Relatively old technology – easier for invasive analysis today
 - TQFP100 package easy to deal with compared to modern alternatives
- Voltage glitching
- Von Neumann architecture, no strong MMU protection
- No possibility for code updates for countermeasure purposes
- Hardware crypto customizations are simple

Strengths

- Key handling and decryption contained within single chip makes it difficult to do key sharing.
- Fast working key change interval (133 ms) makes key sharing difficult.
- No possibility for code update means nowhere to write code for permanent backdoor.
- Internal CLK prevents clock glitching.
- Dead addresses to prevent linear readout of keys.
- Personalization ROM appears to be inaccessible.
- Keys kept only in RAM – must maintain battery backup at all times!
 - Keys appear non-rewritable, preventing cloning units.
- No group keys for EMMs – all unit addressing is to individual units.
 - Must pull keys from a subscribed box in order to get active keys.
- Software appears generally well-designed and written.
- Although DES is used, EMMs are signed using 3 DES keys.
 - Multiple rounds of DES are used, increasing brute force complexity.



Thank you for choosing Pay Per View.

