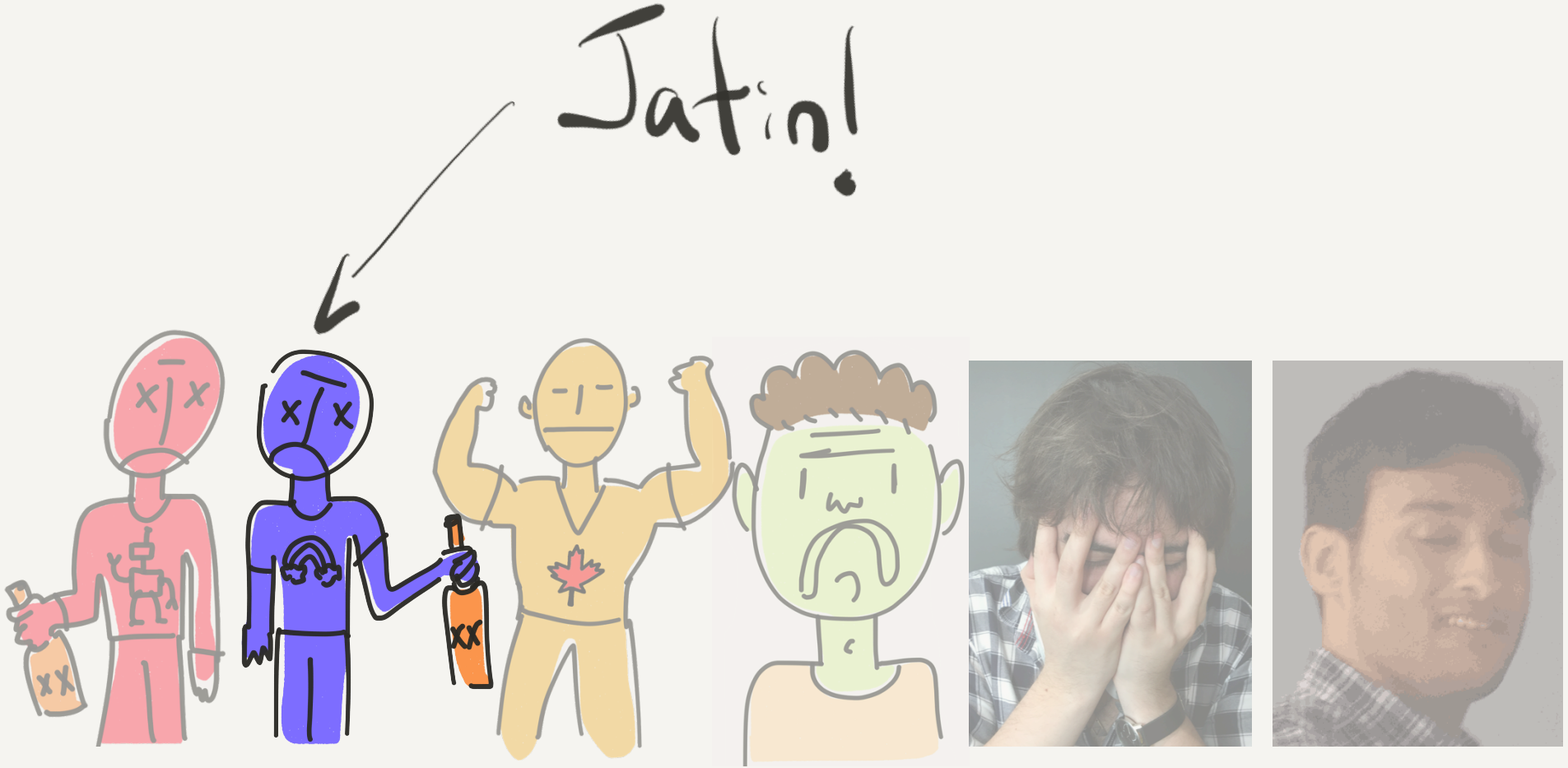

Mystical OSD
Unicorn Demands
Obedience!

https://github.com/RedBalloonShenanigans/MonitorDarkly


A Monitor Darkly
Ang Cui, PhD | Jatin Kataria

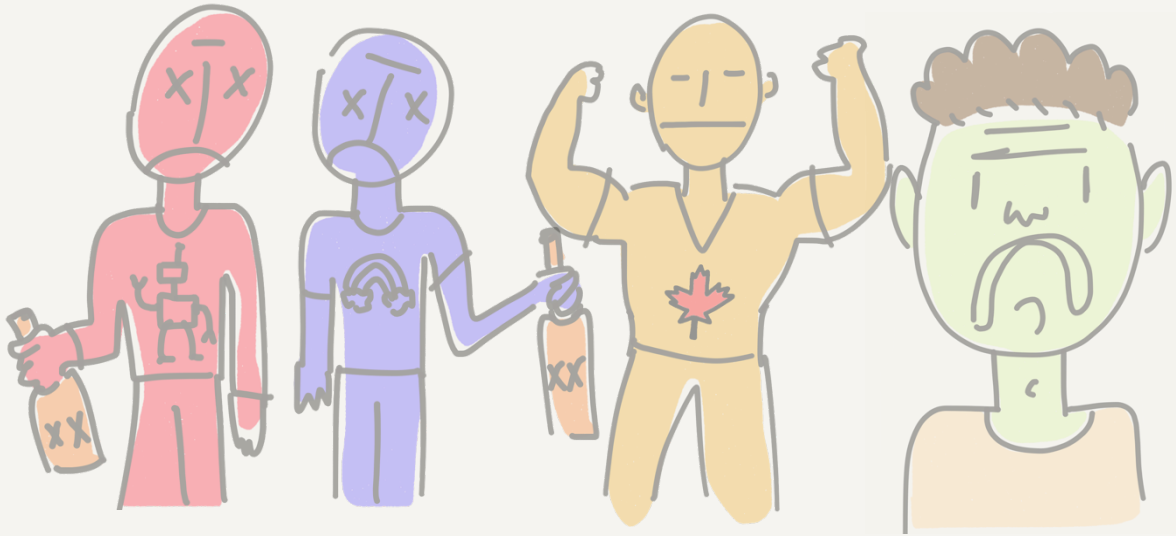

Arg!


Jatin!


strong Francois!


Igor!

Concerned Area Man
(Chris)

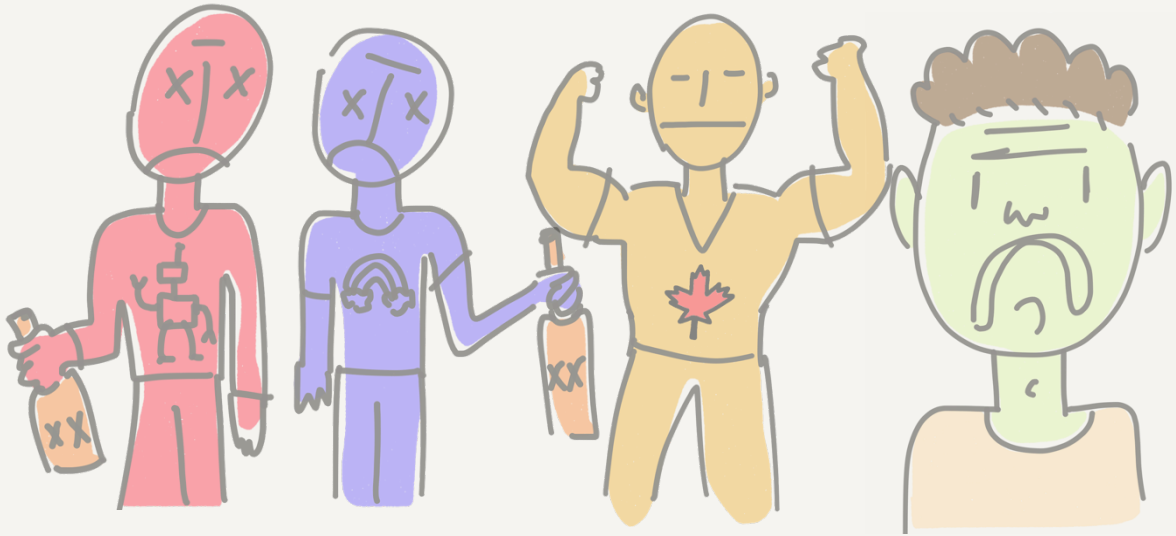






Area Man of Concern

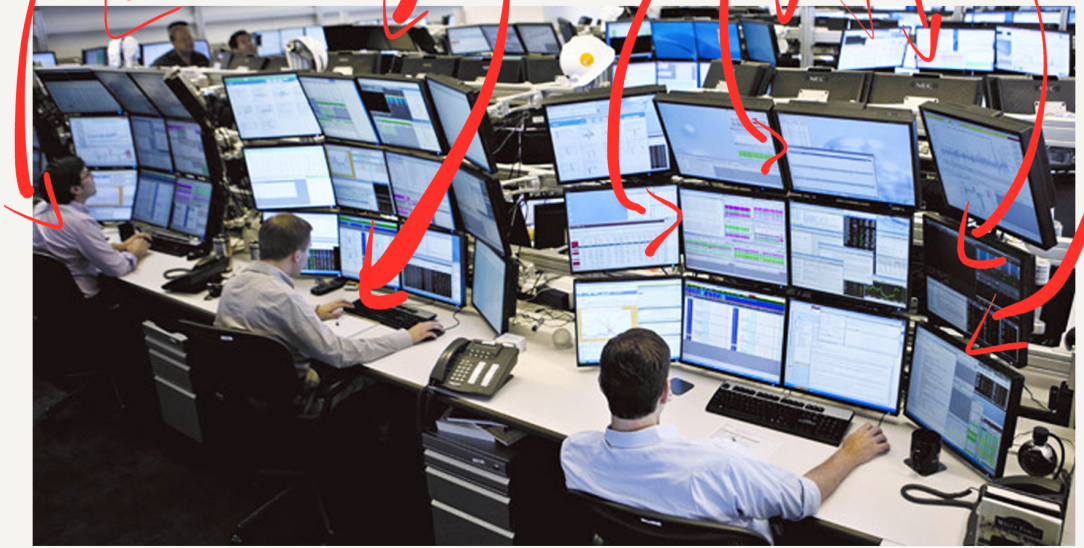
(shakeeb)

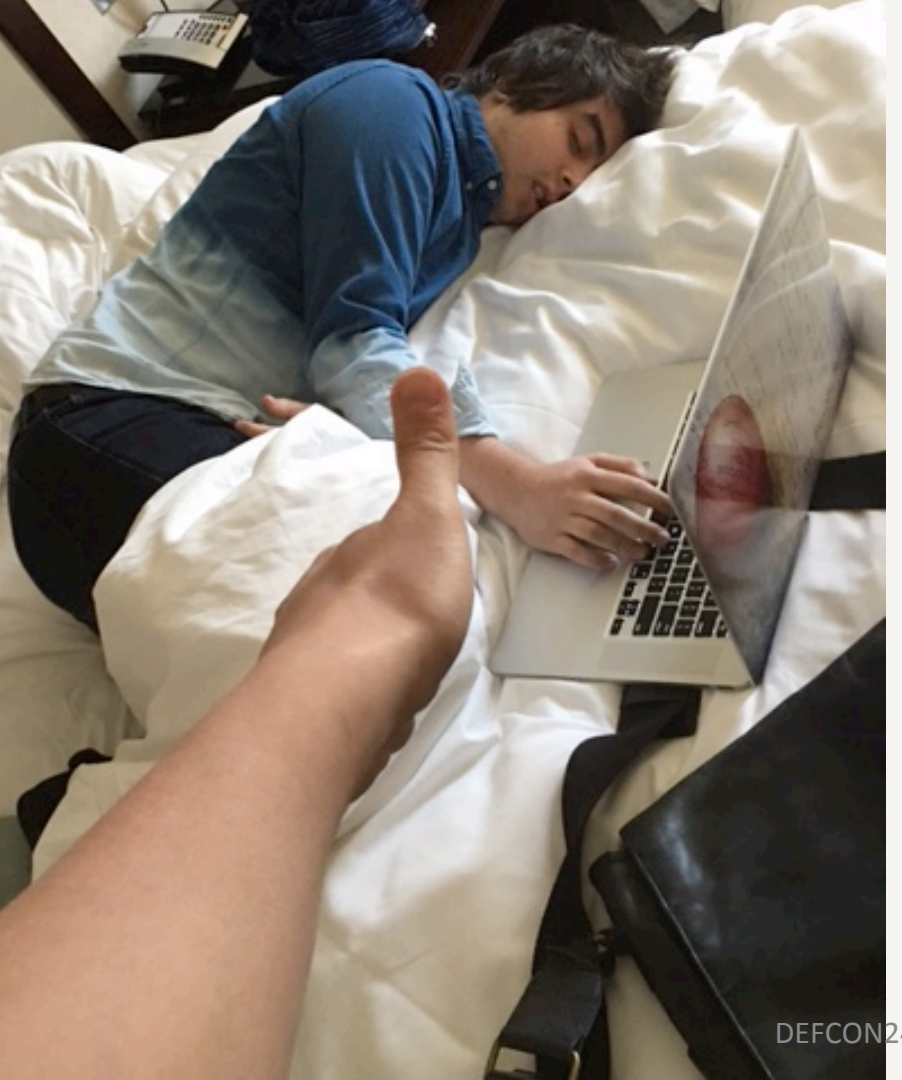




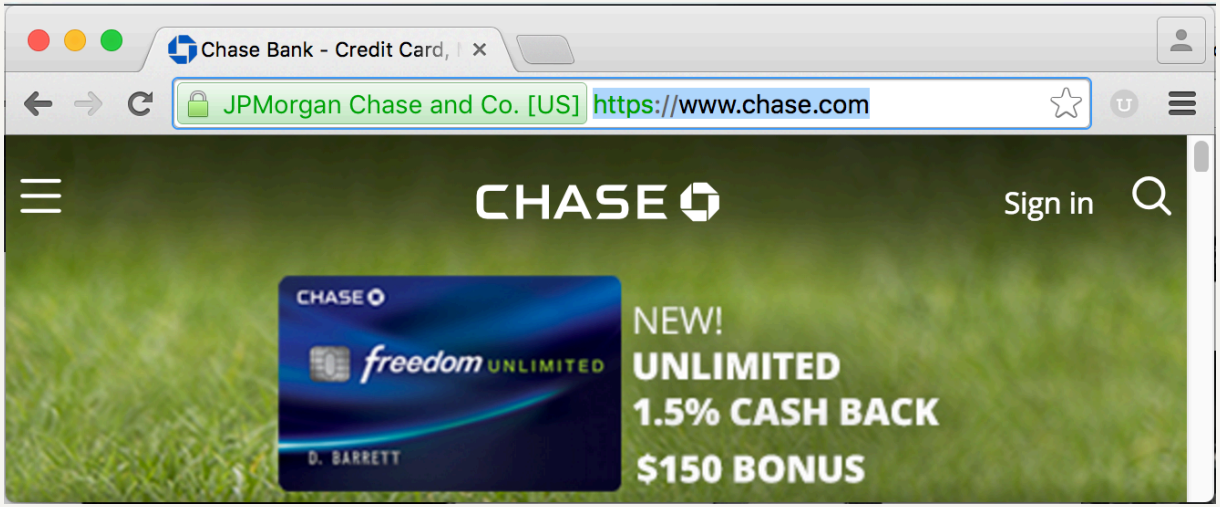


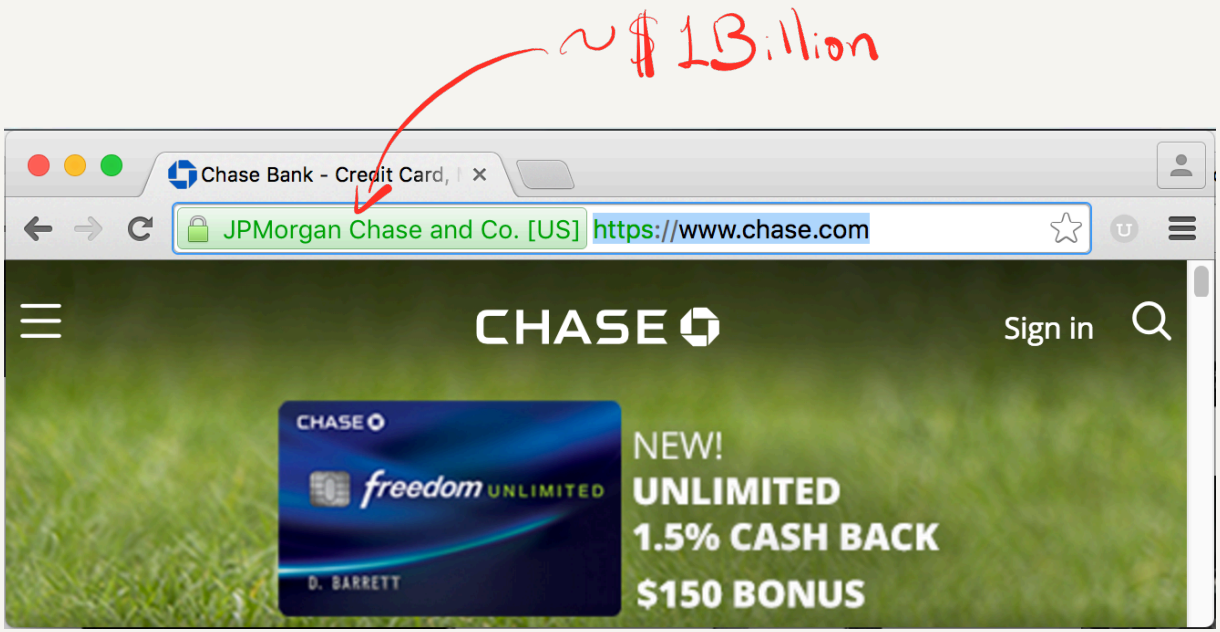
Primary Main Objective !

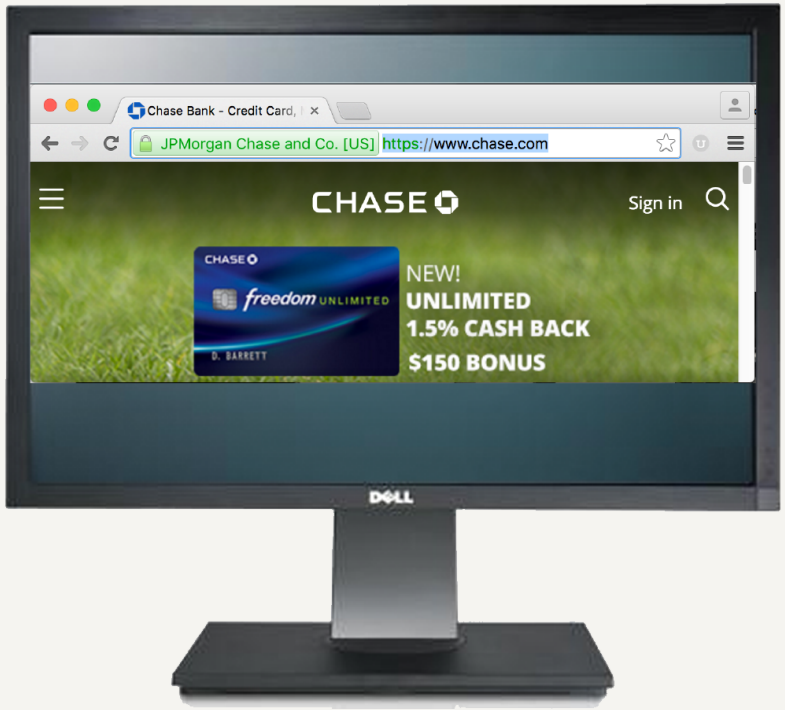




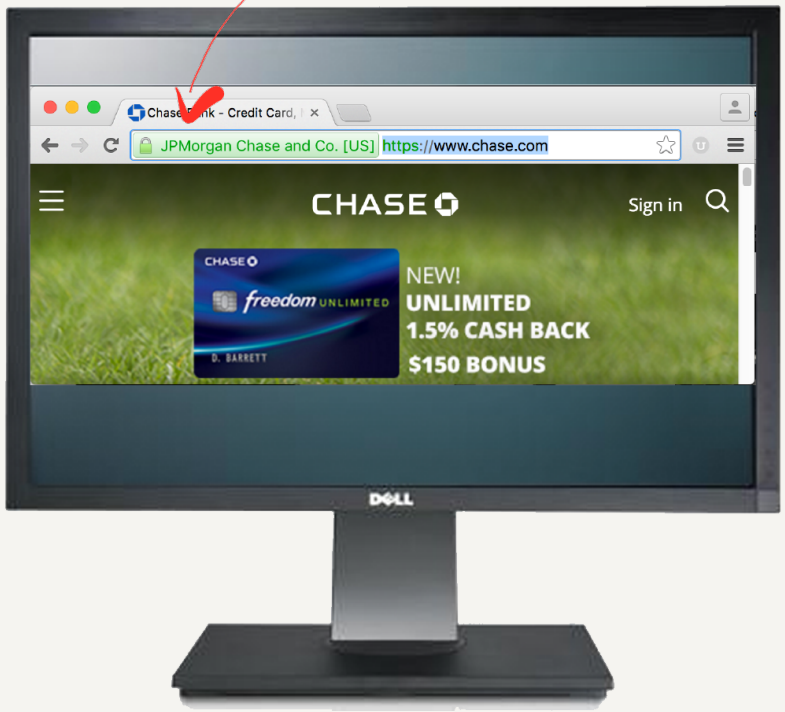
A Good Hacker
IS A
LAZY Hacker
!!


Chase Bank - Credit Card,
JPMorgan Chase and Co. [US]
https://www.chase.com
CHASE
Sign in
CHASE
freedom UNLIMITED
D. BARRETT
NEW!
UNLIMITED
1.5% CASH BACK
$150 BONUS


~$1Billion
Chase Bank - Credit Card,
JPMorgan Chase and Co. [US]
https://www.chase.com
CHASE
Sign in
CHASE
freedom UNLIMITED
D. BARRETT
NEW!
UNLIMITED
1.5% CASH BACK
$150 BONUS


Chase Bank - Credit Card,
JPMorgan Chase and Co. [US]
https://www.chase.com
CHASE
Sign in
freedom UNLIMITED
NEW!
UNLIMITED
1.5% CASH BACK
$150 BONUS
DELL

Min (~$1B, $???)

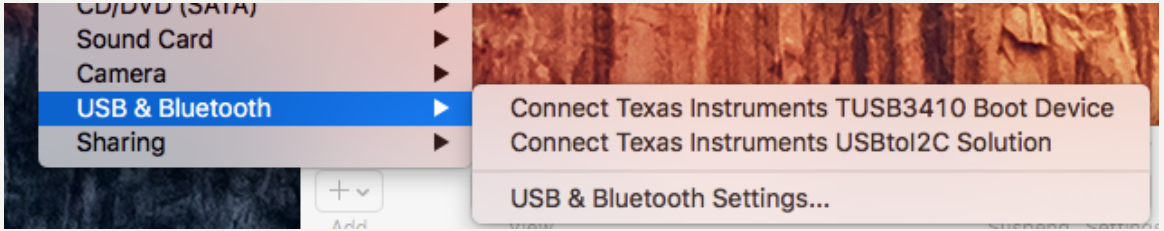

JPMorgan Chase and Co. [US] https://www.chase.com
CHASE
Sign in
NEW!
UNLIMITED
1.5% CASH BACK
$150 BONUS
DELL





whoa, I can haz!




Sound Card
Camera
USB & Bluetooth
Sharing
Connect Texas Instruments TUSB3410 Boot Device
Connect Texas Instruments USBtoI2C Solution
USB & Bluetooth Settings...

← oh hai!

like 1 minute of googling....





Posted by DELL-Chris M on 6 Mar 2015 11:41
Verified by TGMadsen

✔ Verified Answer

Error = No TUSB3410 boot device
Discussion = This does not effect the monitor functionality. It only applies if we decide to do a firmware update
Solution =
* Go here
* Scroll down under Software
* Download/save "TI WDF USBUART Single Driver (Rev. A)" zip file (sllc428a.zip) to your windows desktop
* Unzip the file into its own folder
* Open the folder and run "TI_WDF_USBUART_SINGLE_DRIVER_V6.7.2.0_WHQL.exe"
* It should create the following folder:
C:\Program Files (x86)\Texas Instruments Inc\TI_WDF_USBUART_SINGLE_DRIVER_V6.7.2.0_WHQL
* Run Setup





yo, let's tear down the 34 inch





We both already have sweet monitors.
no one will miss that 34 inch





These Monsters have no heart

There will be no END to their senseless Savagery (and I have a million VIM plugins!)



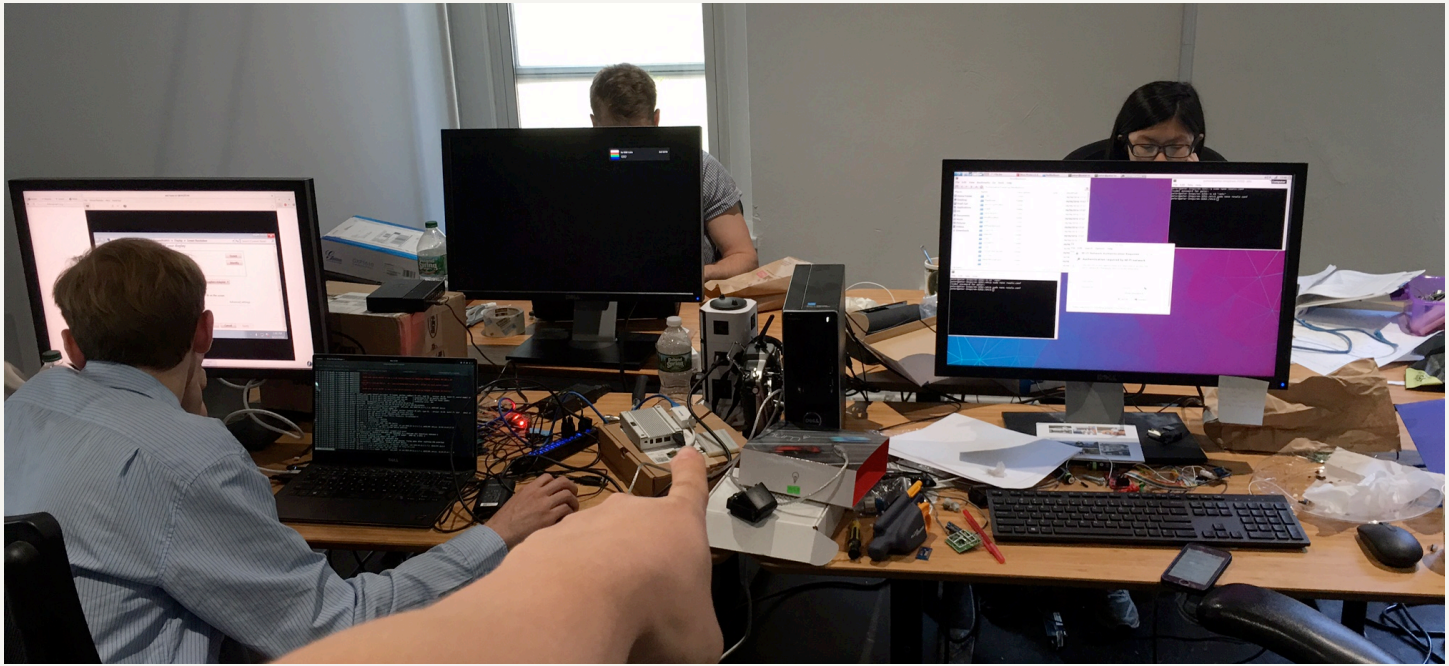
oh gOrsh, how Sad !!





Interns!





Interns get Dell U2410's. So....



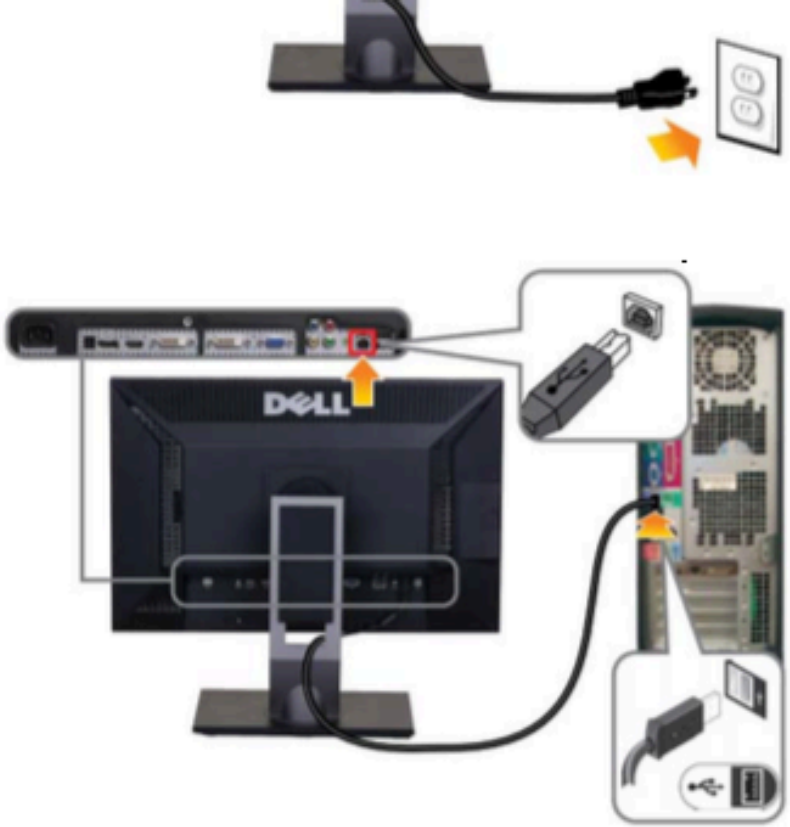
like 15 minutes of googling....

[PDF] DELL U2410 USB FW Upgrade Instruction

https://forum-en.msi.com/index.php?action=dlattach;topic=165660.0;attach...

firmware upgrade.) Desktop & Notebook: 2. Connect the power cord and turn on the. U2410 monitor–to-be flashed. 3. Connect USB uplink cable from the.

3. Connect USB uplink cable from the computer USB port to U2410 USB upstream port.



1. Double click " U2410 ISP TOOL 2.0 " package to start installation ..



U2410 ISP Tool 2.0

Power goes in the Power wall thing!?!



DELL U2410 USB_ISP Ver: 2.0

FW Ver: 90

Browse

Get FW Ver

Start

Exit

J:\DELL U2410_M1F190_0DFF.bin

```
Run CMD_Reset0 successful.

Start writing Driver_file to Scalar.
  Write Driver_file to Scalar successful.

Run CMD_0x500 successful.

MX Flash    ID: c214
Run CMD_Appstest 2 1 successful.

Start writing MONITOR_FW to DDR RAM, Please Wait about 30s.
```

72%

Clear Messages

Open

Look in: Data (H:)

Dell U2410_31F1

Dell U2410_31F170_1B49

Dell U2410_31F172_1B49

My Recent Documents

Desktop

My Documents

My Computer

My Network Places

File name: Dell U2410_31F170_1B49

Files of type: Bin File (*.bin)

DELL U2410 USB_ISP Ver: 2.0

FW Ver: 90

Browse

Get FW Ver

Start

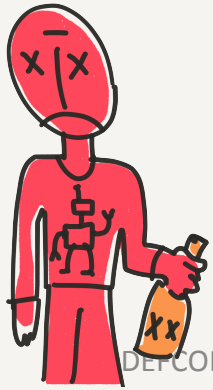
Exit

J:\DELL U2410_M1F190_0DFF.bin

```
Run CMD_Reset0 successful.

Start writing Driver_file to Scalar.
  Write Driver_file to Scalar successful.

Run CMD_0x500 successful.

MX Flash    ID: c214
Run CMD_Appstest 2 1 successful.

Start writing MONITOR_FW to DDR RAM, Please Wait about 30s.
  Writing MONITOR_FW to DDR RAM fail!

Cycle AC power to your monitor now .....
```

12%

Clear Messages



DEFCON2

DELL U2410 USB_ISP Ver: 2.0

FW Ver: XX

Browse

Get FW Ver

Start

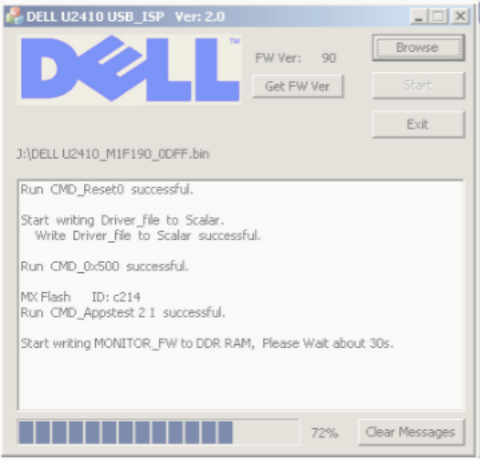
Exit

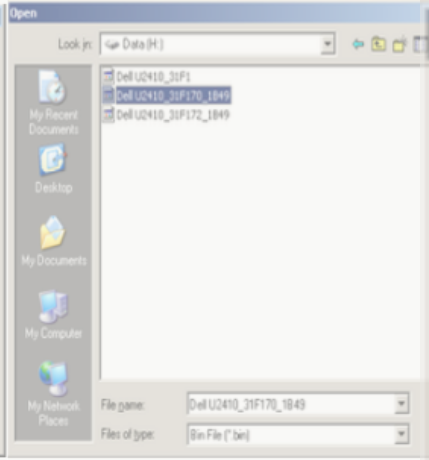
J:\DELL U2410_M1F190_0DFF.bin

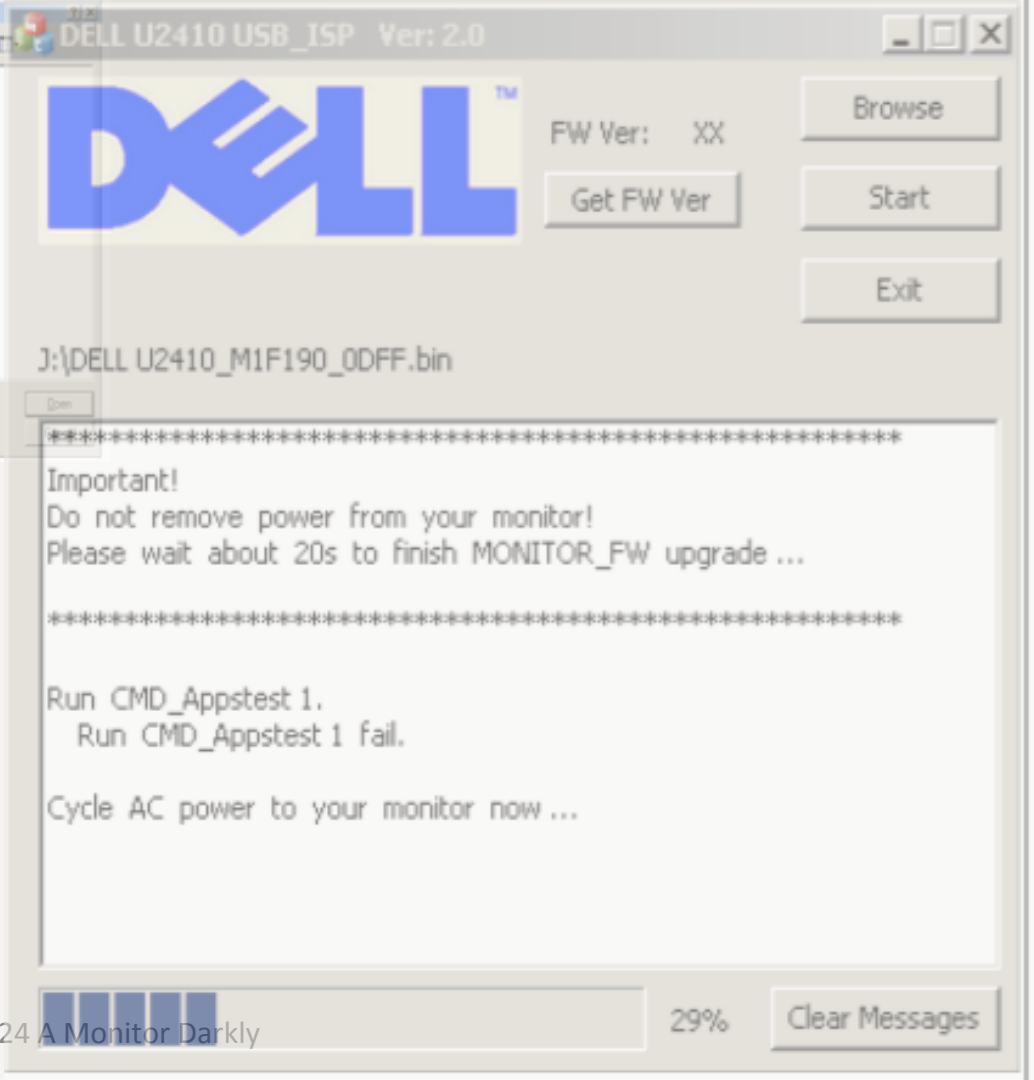
```
**********************************************************
Important!
Do not remove power from your monitor!
Please wait about 20s to finish MONITOR_FW upgrade ...

**********************************************************

Run CMD_Appstest 1.
  Run CMD_Appstest 1 fail.

Cycle AC power to your monitor now ...
```

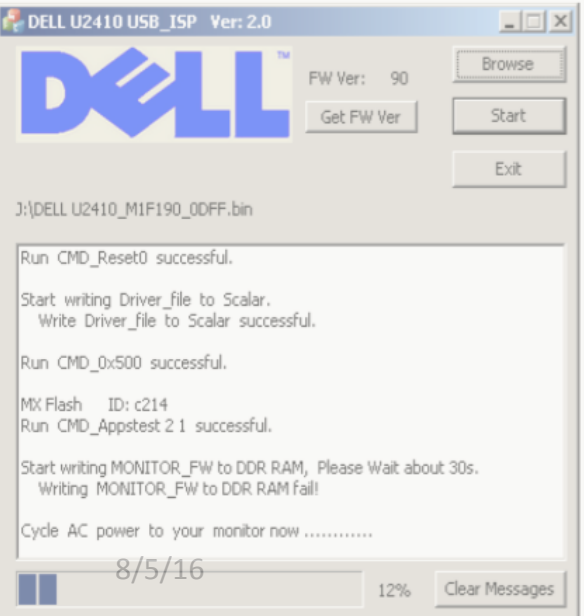
29%

Clear Messages


DELL U2410 USB_ISP Ver: 2.0
FW Ver: 90
Browse
Get FW Ver
Start
Exit
J:\DELL U2410_M1F190_0DFF.bin
Run CMD_Reset0 successful.
Start writing Driver_file to Scalar.
Write Driver_file to Scalar successful.
Run CMD_0x500 successful.
MX Flash ID: c214
Run CMD_Appstest 2 1 successful.
Start writing MONITOR_FW to DDR RAM, Please Wait about 30s.
72%
Clear Messages


Open
Look in: Data (H:)
Dell U2410_31F1
Dell U2410_31F170_1849
Dell U2410_31F172_1849
My Recent Documents
Desktop
My Documents
My Computer
My Network Places
File name: Dell U2410_31F170_1849
Files of type: Bin File (*.bin)


DELL U2410 USB_ISP Ver: 2.0
FW Ver: XX
Browse
Get FW Ver
Start
Exit
J:\DELL U2410_M1F190_0DFF.bin
****
Important!
Do not remove power from your monitor!
Please wait about 20s to finish MONITOR_FW upgrade ...
****
Run CMD_Appstest 1.
Run CMD_Appstest 1 fail.
Cycle AC power to your monitor now ...
29%
Clear Messages

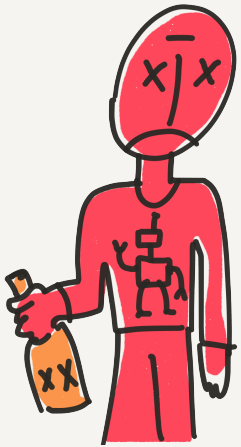

DELL U2410 USB_ISP Ver: 2.0
FW Ver: 90
Browse
Get FW Ver
Start
Exit
J:\DELL U2410_M1F190_0DFF.bin
Run CMD_Reset0 successful.
Start writing Driver_file to Scalar.
Write Driver_file to Scalar successful.
Run CMD_0x500 successful.
MX Flash ID: c214
Run CMD_Appstest 2 1 successful.
Start writing MONITOR_FW to DDR RAM, Please Wait about 30s.
Writing MONITOR_FW to DDR RAM fail!
Cycle AC power to your monitor now .....
12%
Clear Messages


app test?



Genesis?



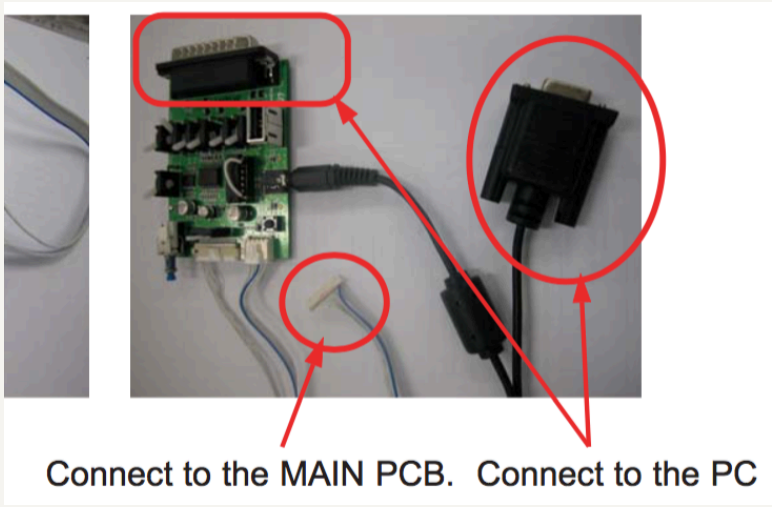


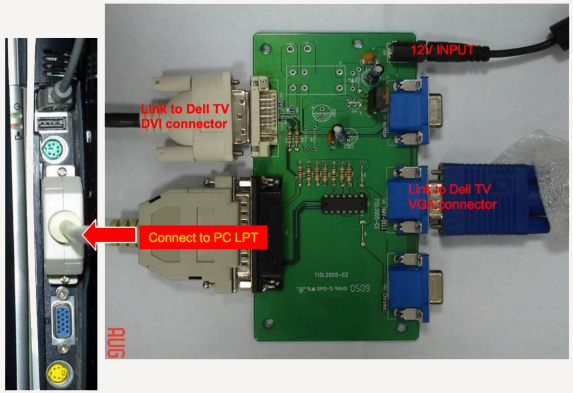
app test?

GProbe?

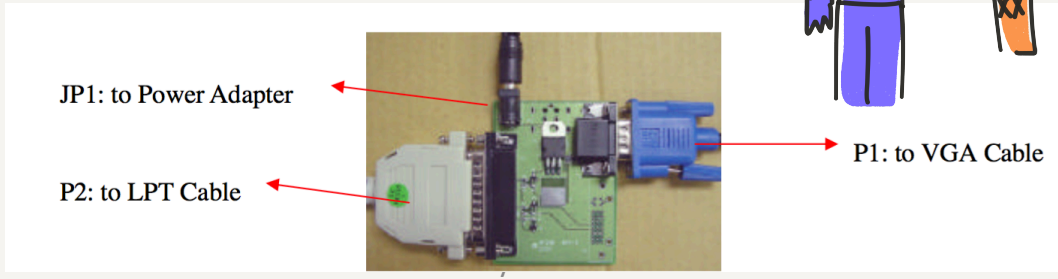


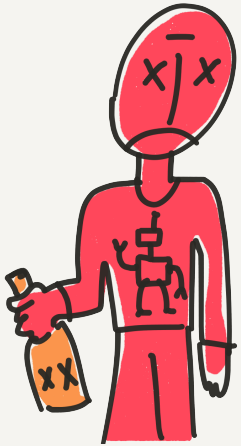

google
google
google
google
google
google
app test?


Connect to the MAIN PCB.
Connect to the PC


12V INPUT
Link to Dell TV DVI connector
Link to Dell TV VGA connector
Connect to PC LPT

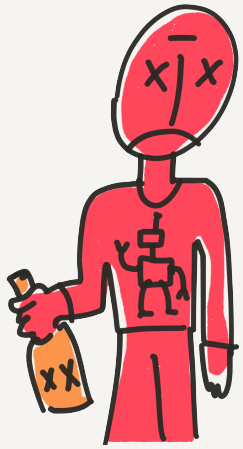



JP1: to Power Adapter
P2: to LPT Cable
P1: to VGA Cable



st micro?





stmicro?

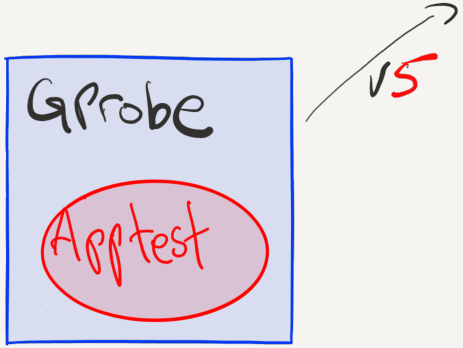
Innolux?

Athena?

Dell?




Apptest



GENESIS
DISPLAY PERFECTION

Genesis Microchip Technical Documentation

GProbe_User_Guide

5.2.1.15 RunCode

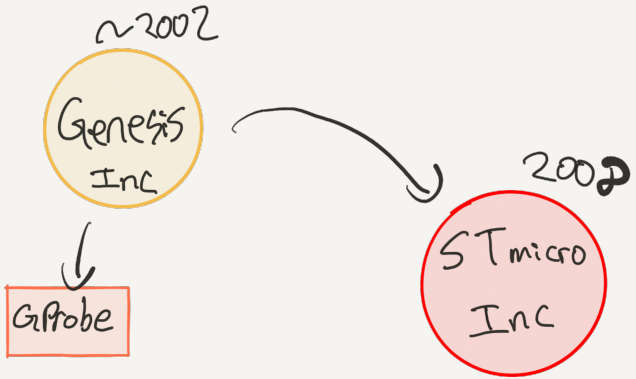

google
google
google
gooogle
google


~2002
Genesis
Inc

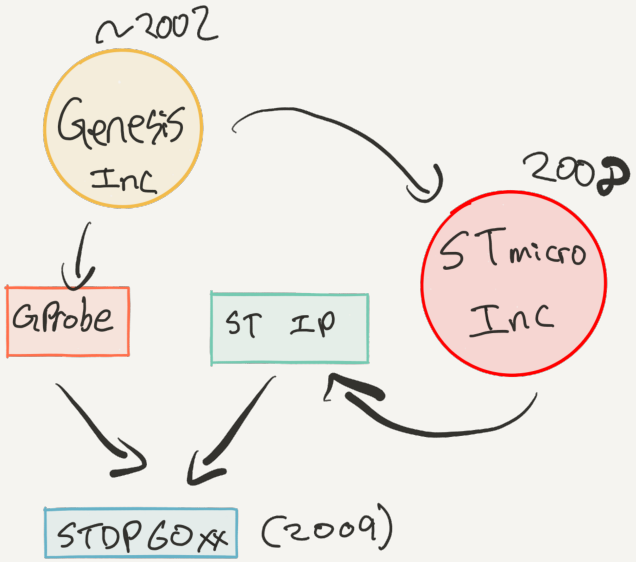



~2002
Genesis
Inc
GProbe

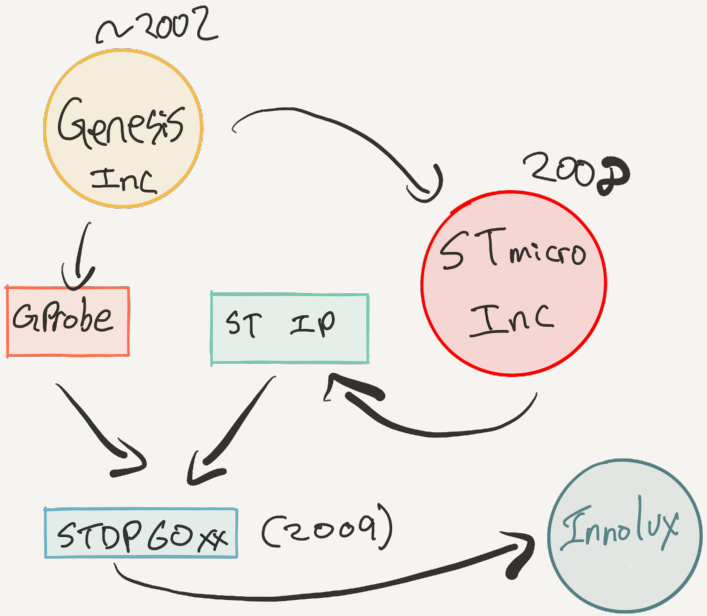



~2002
Genesis Inc
GProbe
2002
STmicro Inc

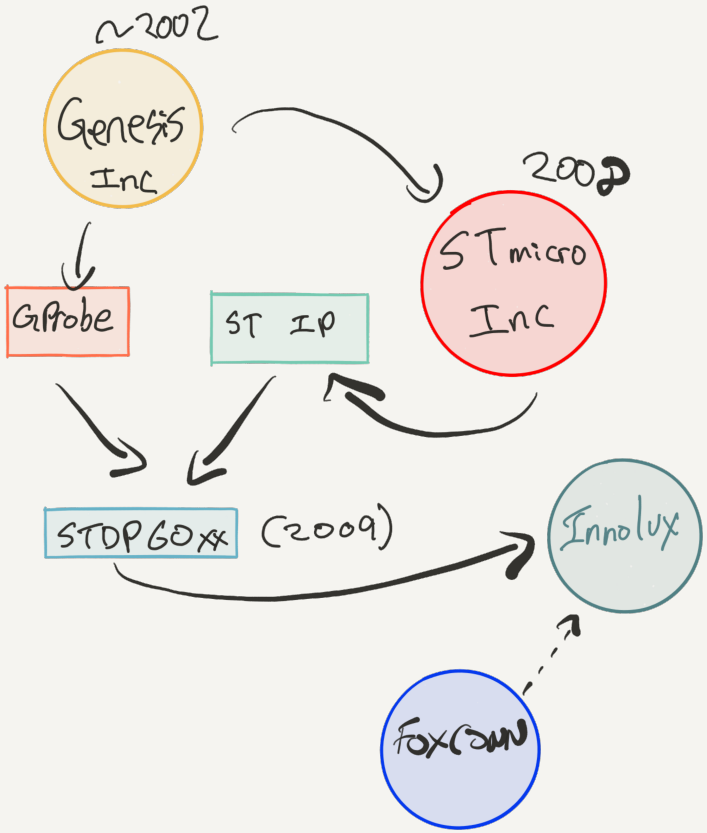



~2002
Genesis Inc
2002
STmicro Inc
GPrObe
ST IP
STOPGOXX
(2009)

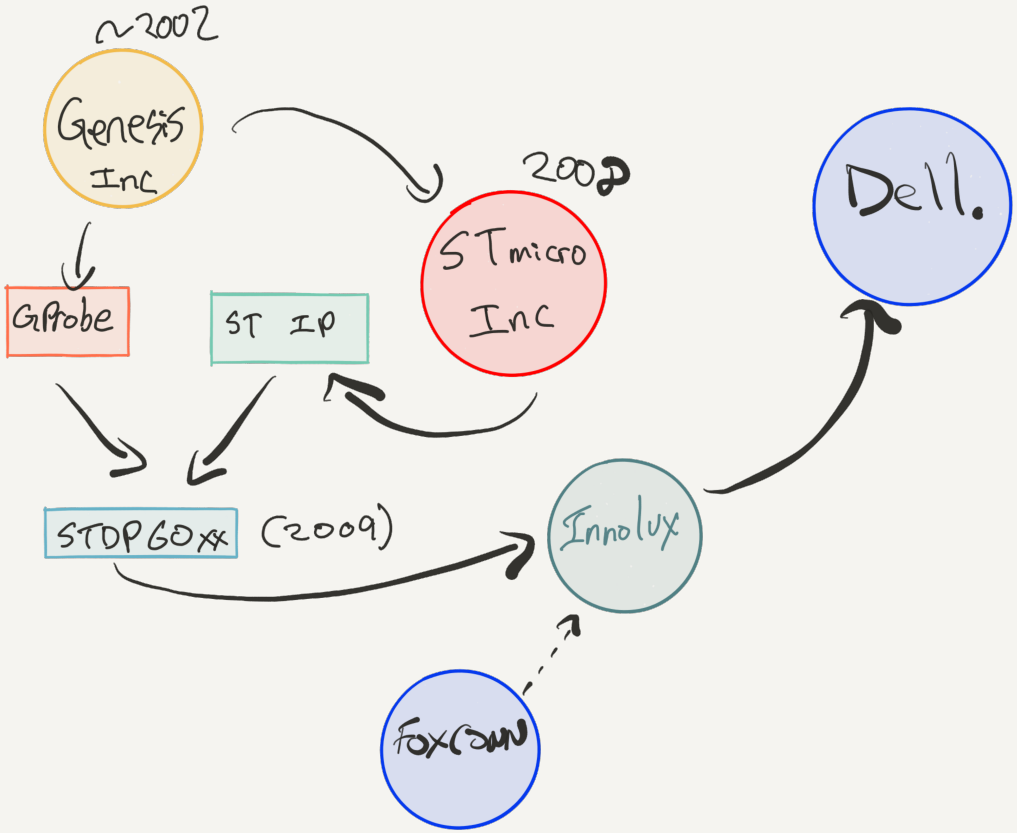



~2002
Genesis Inc
2002
STmicro Inc
Gprobe
ST IP
STOPGOxx
(2009)
Innolux

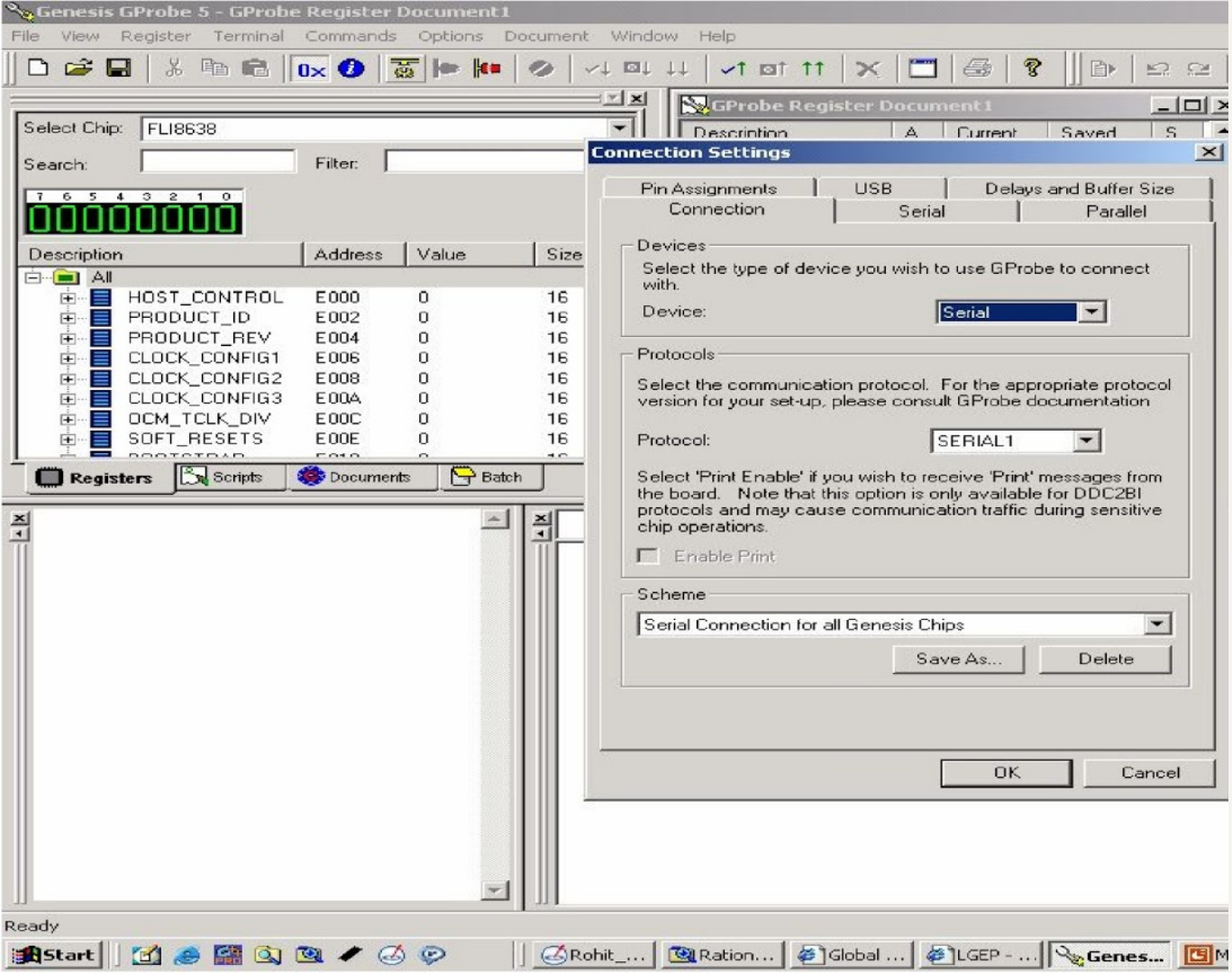


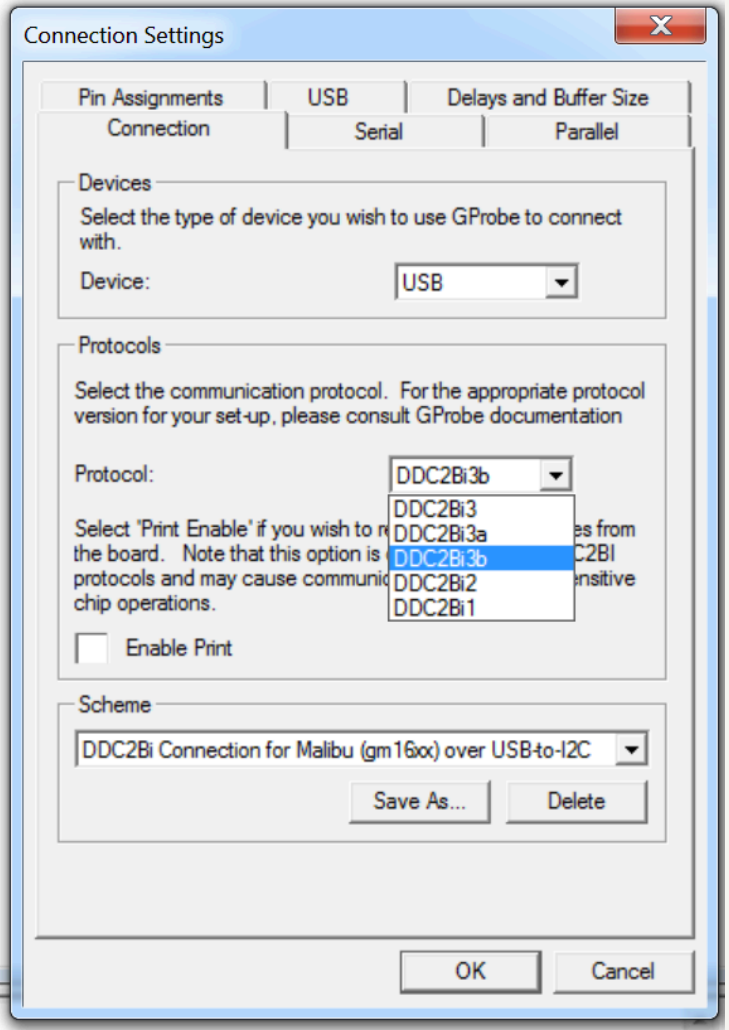

~2002
Genesis Inc
2002
STmicro Inc
Gprobe
ST IP
STOPGOxx (2009)
Innolux
FOXCONN




~2002
Genesis Inc
2002
STmicro Inc
Dell.
Gprobe
ST IP
STDP600
(2009)
Innolux
Foxconn




Genesis GProbe 5 - GProbe Register Document1
File View Register Terminal Commands Options Document Window Help
Select Chip: FLI8638
Search:
Filter:
7 6 5 4 3 2 1 0
00000000
Description Address Value Size
All
HOST_CONTROL E000 0 16
PRODUCT_ID E002 0 16
PRODUCT_REV E004 0 16
CLOCK_CONFIG1 E006 0 16
CLOCK_CONFIG2 E008 0 16
CLOCK_CONFIG3 E00A 0 16
OCM_TCLK_DIV E00C 0 16
SOFT_RESETS E00E 0 16
Registers
Scripts
Documents
Batch
GProbe Register Document1
Description
Current
Saved
Connection Settings
Pin Assignments
USB
Delays and Buffer Size
Connection
Serial
Parallel
Devices
Select the type of device you wish to use GProbe to connect with.
Device:
Serial
Protocols
Select the communication protocol. For the appropriate protocol version for your set-up, please consult GProbe documentation
Protocol:
SERIAL1
Select 'Print Enable' if you wish to receive 'Print' messages from the board. Note that this option is only available for DDC2BI protocols and may cause communication traffic during sensitive chip operations.
Enable Print
Scheme
Serial Connection for all Genesis Chips
Save As...
Delete
OK
Cancel
Ready
Start
Rohit_...
Ration...
Global ...
LGEP - ...
Genes...

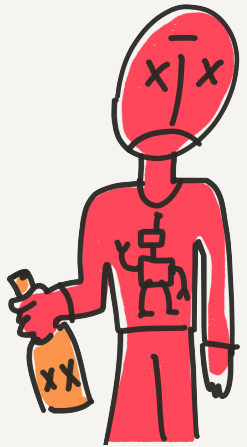

Connection Settings
Pin Assignments
USB
Delays and Buffer Size
Connection
Serial
Parallel
Devices
Select the type of device you wish to use GProbe to connect with.
Device:
USB
Protocols
Select the communication protocol. For the appropriate protocol version for your set-up, please consult GProbe documentation
Protocol:
DDC2Bi3b
DDC2Bi3
DDC2Bi3a
DDC2Bi3b
DDC2Bi2
DDC2Bi1
Select 'Print Enable' if you wish to r
es from
the board. Note that this option is
C2BI
protocols and may cause communic
nsitive
chip operations.
Enable Print
Scheme
DDC2Bi Connection for Malibu (gm16xx) over USB-to-I2C
Save As...
Delete
OK
Cancel

220 25.856425	1.2	host	USBMS	40	SCSI: Response LUN: 0x00 (CDB:0
221 23.856425	host	1.2	USBMS	58	SCSI Command: 0xcf LUN:0x00
222 23.856425	host	1.2	USB	539	URB_BULK out
223 24.028286	1.2	host	USBMS	40	SCSI: Response LUN: 0x00 (CDB:0
224 24.028286	host	1.2	USBMS	58	SCSI: Test Unit Ready LUN: 0x00
225 24.028286	1.2	host	USBMS	40	SCSI: Response LUN: 0x00 (Test
226 24.028286	host	1.2	USBMS	58	SCSI: Request Sense LUN: 0x00
227 24.043910	1.2	host	USB	45	URB_BULK in
228 24.043910	1.2	host	USBMS	40	SCSI: Response LUN: 0x00 (Reque
229 24.043910	host	1.2	USBMS	58	SCSI Command: 0xcf LUN:0x00
230 24.075157	1.2	host	USB	285	URB_BULK in
231 24.075157	1.2	host	USBMS	40	SCSI: Response LUN: 0x00 (CDB:0
232 24.996955	host	1.2	USBMS	58	SCSI: Test Unit Ready LUN: 0x00
233 24.996955	1.2	host	USBMS	40	SCSI: Response LUN: 0x00 (Test
234 24.996955	host	1.2	USBMS	58	SCSI: Request Sense LUN: 0x00
235 24.996955	1.2	host	USB	45	URB_BULK in
236 24.996955	1.2	host	USBMS	40	SCSI: Response LUN: 0x00 (Reque
237 25.121945	host	1.2	USBMS	58	SCSI Command: 0xcf LUN:0x00
238 25.137569	host	1.2	USB	539	URB_BULK out
239 25.137569	1.2	host	USBMS	40	SCSI: Response LUN: 0x00 (CDB:0
240 25.137569	host	1.2	USBMS	58	SCSI Command: 0xcf LUN:0x00
241 25.153192	1.2	host	USB	285	URB_BULK in
242 25.153192	1.2	host	USBMS	40	SCSI: Response LUN: 0x00 (CDB:0
243 25.153192	host	1.2	USBMS	58	SCSI Command: 0xcf LUN:0x00
244 25.153192	host	1.2	USB	539	URB_BULK out
245 25.387548	1.2	host	USBMS	40	SCSI: Response LUN: 0x00 (CDB:0
246 25.387548	host	1.2	USBMS	58	SCSI Command: 0xcf LUN:0x00
247 25.403172	1.2	host	USB	285	URB_BULK in
248 25.418795	1.2	host	USBMS	40	SCSI: Response LUN: 0x00 (CDB:0
249 25.965625	host	1.2	USBMS	58	SCSI Command: 0xcf LUN:0x00

```
▷ Frame 238: 539 bytes on wire (4312 bits), 539 bytes captured (4312 bits)
▽ USB URB
    USBPcap pseudoheader length: 27
    IRP ID: 0xffffffffaa675998
    IRP USBD_STATUS: USBD_STATUS_SUCCESS (0x00000000)
    URB Function: URB_FUNCTION_BULK_OR_INTERRUPT_TRANSFER (0x0009)
  ▷ IRP information: 0x00, Direction: FDO -> PDO
    URB bus id: 2
    Device address: 1
```

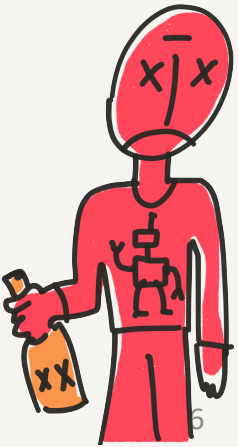
```
0000  1b 00 98 59 67 aa ff ff  ff ff 00 00 00 00 09 00   ...Yg... ........
0010  00 02 00 01 00 02 03 00  02 00 00 01 6e c0 00 09   ........ ....n...
0020  f9 51 86 c2 00 00 04 20  00 5f cc cc cc cc cc cc   .Q.....  ._......
0030  cc cc cc cc cc cc cc cc  cc cc cc cc cc cc cc cc   ........ ........
0040  cc cc cc cc cc cc cc cc  cc cc cc cc cc cc cc cc   ........ ........
0050  cc cc cc cc cc cc cc cc  cc cc cc cc cc cc cc cc   ........ ........
0060  cc cc cc cc cc cc cc cc  cc cc cc cc cc cc cc cc   ........ ........
0070  cc cc cc cc cc cc cc cc  cc cc cc cc cc cc cc cc   ........ ........
0080  cc cc cc cc cc cc cc cc  cc cc cc cc cc cc cc cc   ........ ........
0090  cc cc cc cc cc cc cc cc  cc cc cc cc cc cc cc cc   ........ ........
```

USB Capture of Fw Update



A brief story about

Display Data Channel



VESA®

Display Data Channel Command Interface (DDC/CI) Standard

Video Electronics Standards Association

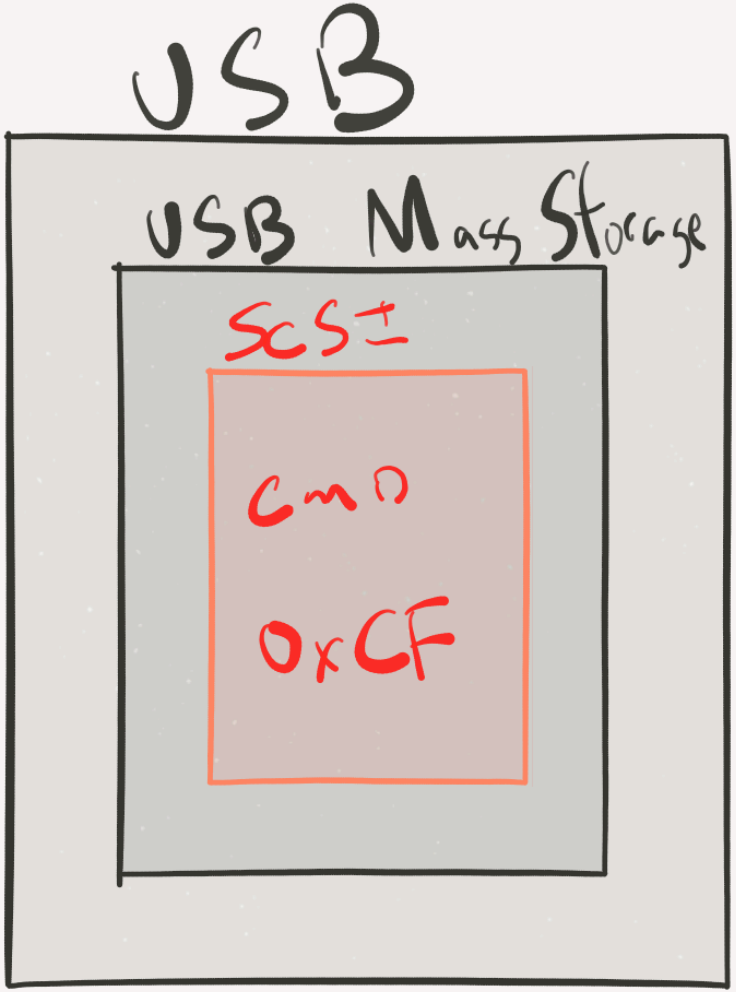
2150 North First Street, Suite 440
San Jose, CA 95131-2029

Phone: (408) 435-0333
Fax: (408) 435-8225

This table summarizes the DDC level upgrade requirements for both the Graphic Host and the Display Device.

From	To	Graphic Host H/W upgrade	Graphic Host S/W upgrade	Display Monitor H/W upgrade	Display Monitor S/W upgrade
DDC1	DDC2B	I^2C Bus Single Master (2 I/Os)	BIOS	I^2C Slave address A0/A1 support	DDC2B driver
DDC2B	DDC2Bi	No upgrade	DDC.DLL driver	6E/6F Slave address support	DDC2Bi driver (Simplified Access Bus)
DDC2B	DDC2B+	50/51 Slave address support	Access Bus Host driver (single device)	6E/6F Slave address support	Access Bus driver
DDC2B	DDC2AB	50/51 Slave address support	Access Bus Host driver (full spec)	6E/6F Slave address support	Access Bus driver

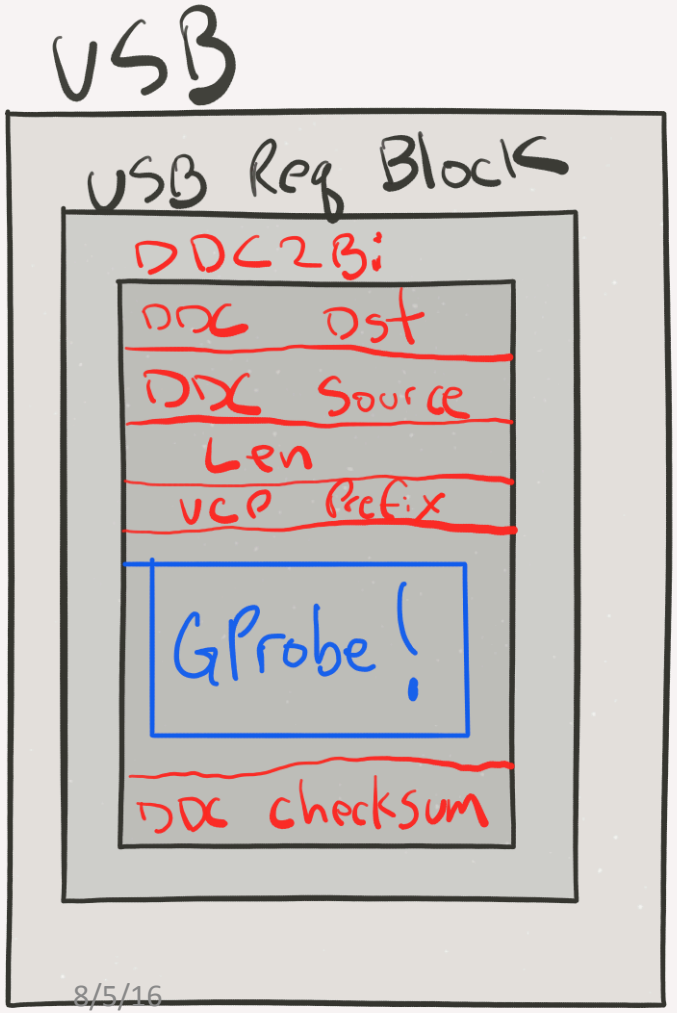
HTTP://CAXAPA.RU/THUMBS/349020/DDCCIV1.PDF



INITIATE COMMUNICATION

0XCF = VENDOR SPECIFIC SCSI CMD





ENCAPSULATED DDC2Bi PACKET

Sample GProbe Packet

RunCode

Request			
Name	Length	Value	Comment
Length	1	Length of message + 1	
Command	1	0x1D	
Address	2		
Response			
ACK/NACK			

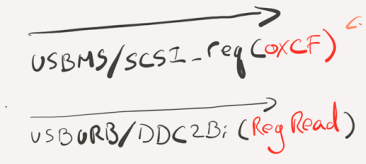


Packet from PC to Genesis chip			
Name	**Length**	**Value**	**Comment**
DDC Destination	1	0x6E	
DDC Source	1	0x51	
Length	1	0x80 | Length of VCP Prefix + Message	Please note that the length of VCP Prefix + Message cannot exceed 127.
VCP Prefix	3	0xC2 0x00 0x00	
Message	variable		

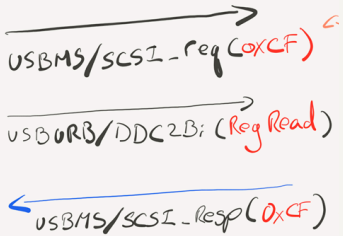
Checksum	1	XOR of all previous bytes	byte chksum = 0;
			for (i = 0; i < buffer_size; ++i)
			chksum ^= buf[i];

DDC2Bi Packet & Checksum Algorithm





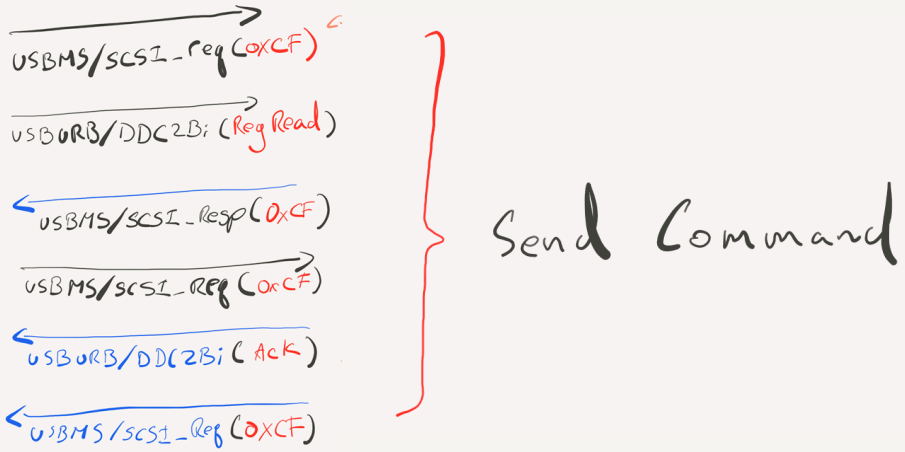
A SIMPLE REQUEST

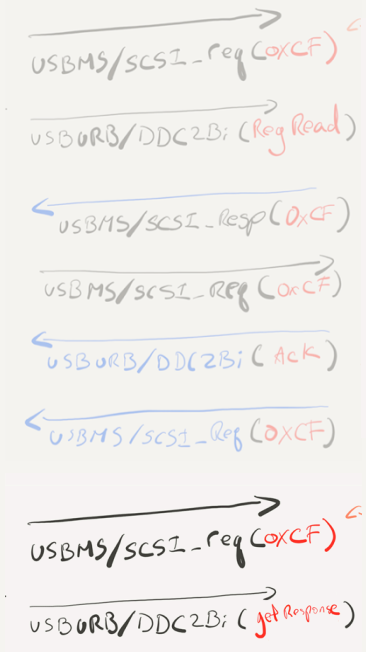


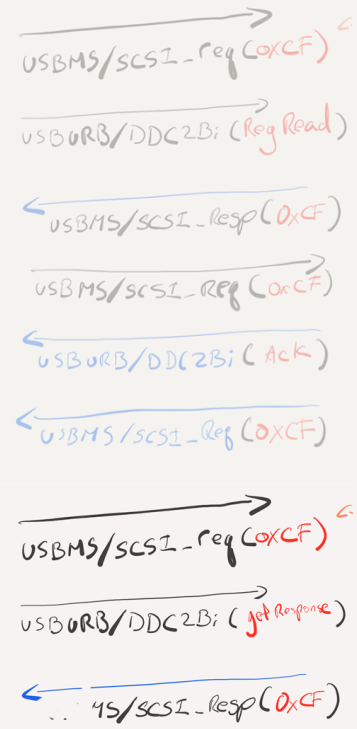
A SIMPLE REQUEST



A SIMPLE REQUEST


USBMS/SCSI_req (0xCF)
USBURB/DDC2Bi (Req Read)
USBMS/SCSI_Resp (0xCF)
USBMS/SCSI_Req (0xCF)
USBURB/DDC2Bi (Ack)
USBMS/SCSI_Req (0xCF)
Send Command


USBMS/SCSI_req(0xCF)
USBURB/DDC2Bi(Req Read)
USBMS/SCSI_Resp(0xCF)
USBMS/SCSI_Req(0xCF)
USBURB/DDC2Bi(Ack)
USBMS/SCSI_Req(0xCF)
USBMS/SCSI_req(0xCF)
USBURB/DDC2Bi(Get Response)
Send Command

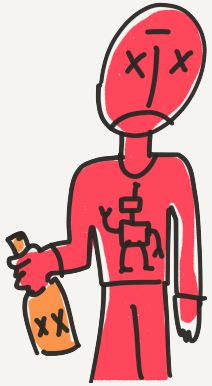


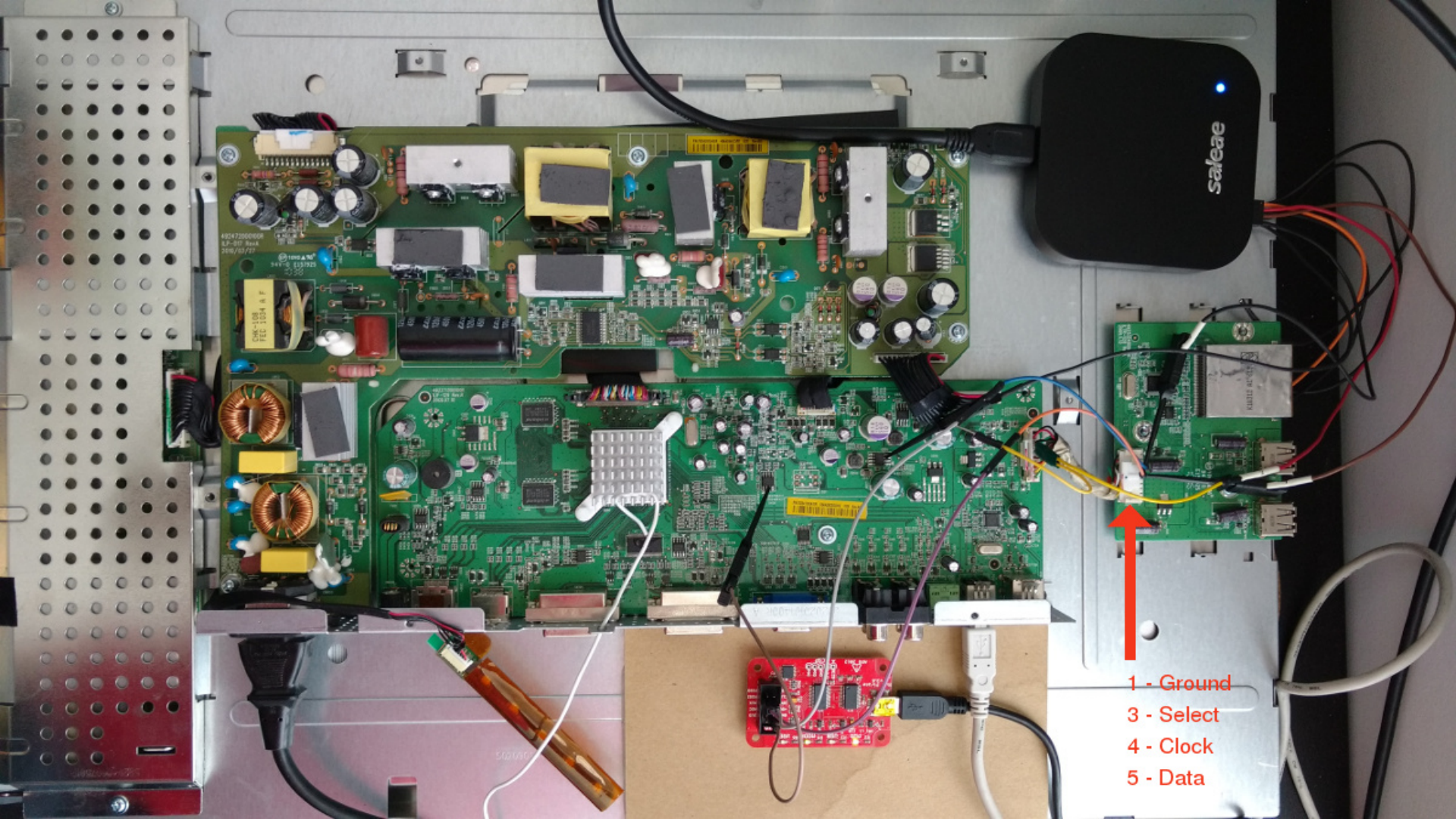
Send Command

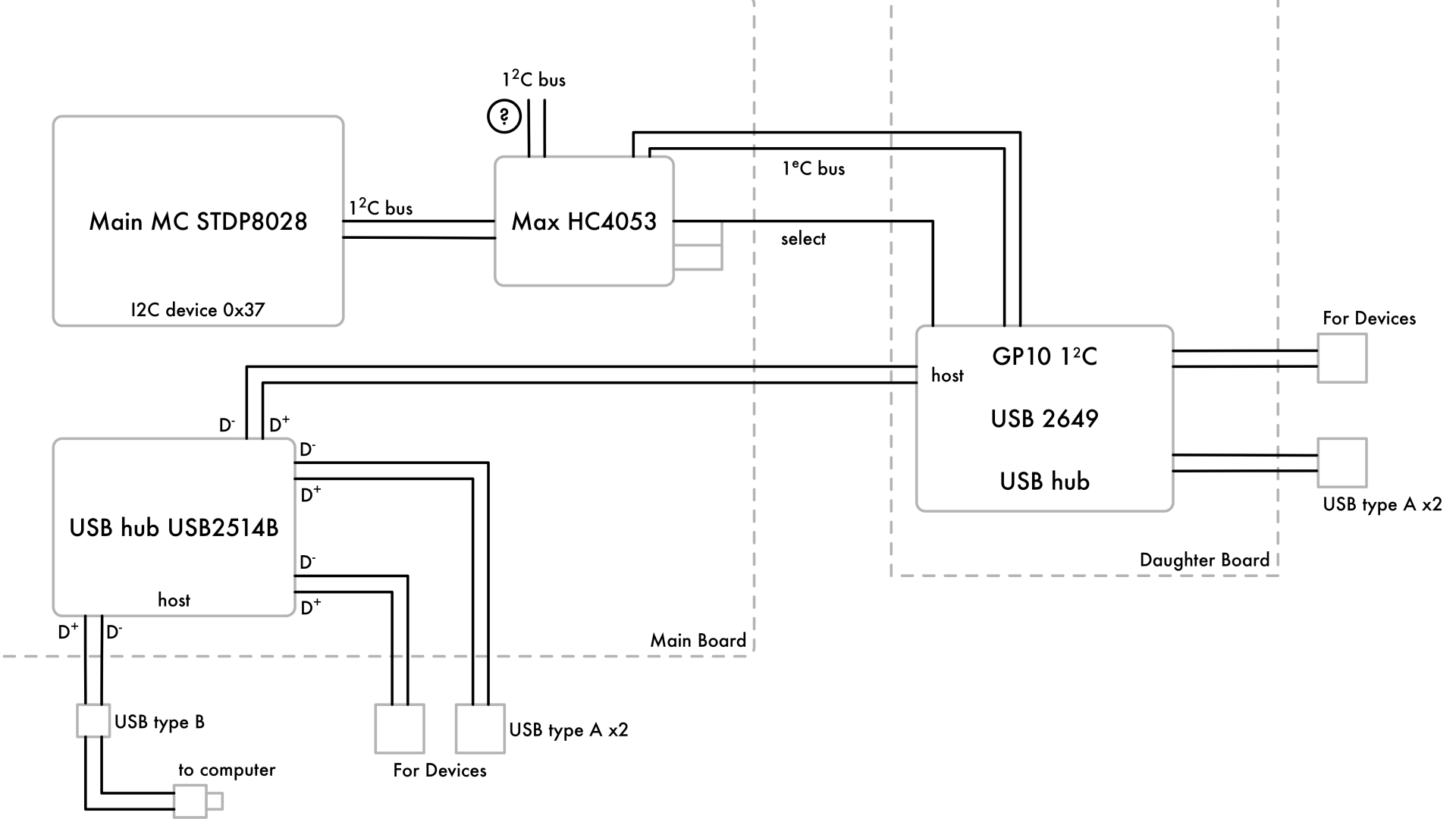

USBMS/SCSI_req(0xCF)
USBURB/DDC2Bi(Req Read)
USBMS/SCSI_Resp(0xCF)
USBMS/SCSI_Req(0xCF)
USBURB/DDC2Bi(Ack)
USBMS/SCSI_Req(0xCF)
USBMS/SCSI_req(0xCF)
USBURB/DDC2Bi(Get Response)
MS/SCSI_Resp(0xCF)
USBMS/SCSI_Req(0xCF)
Send Command

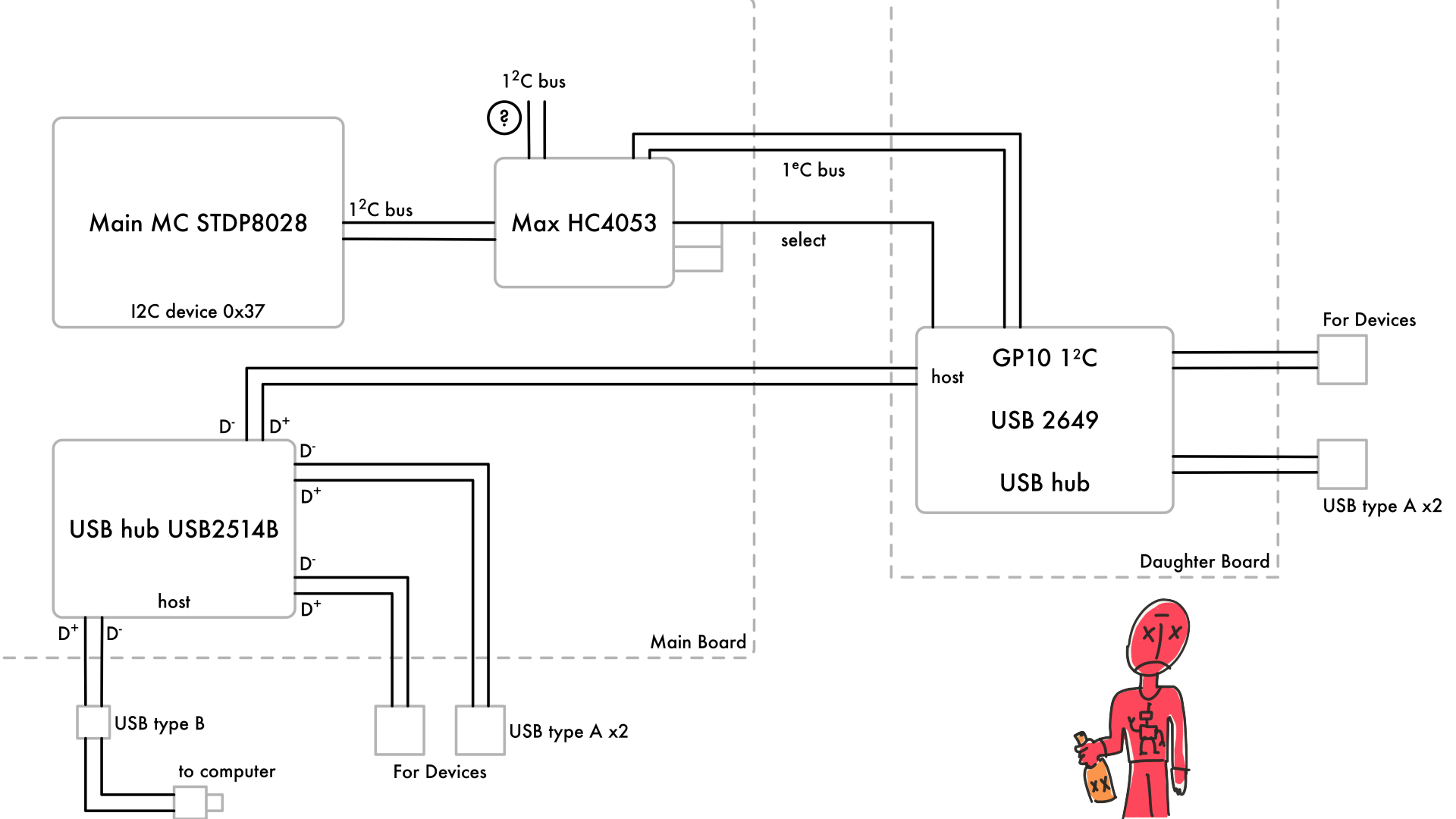

USBMS/SCSI_Req (0xCF)
USBURB/DDC2Bi (Req Read)
USBMS/SCSI_Resp (0xCF)
USBMS/SCSI_Req (0xCF)
USBURB/DDC2Bi (Ack)
USBMS/SCSI_Req (0xCF)
Send Command
USBMS/SCSI_Req (0xCF)
USBURB/DDC2Bi (Get Response)
MS/SCSI_Resp (0xCF)
USBMS/SCSI_Req (0xCF)
USBURB/DDC2Bi (Response)
USBMS/SCSI_Req (0xCF)
Get Result

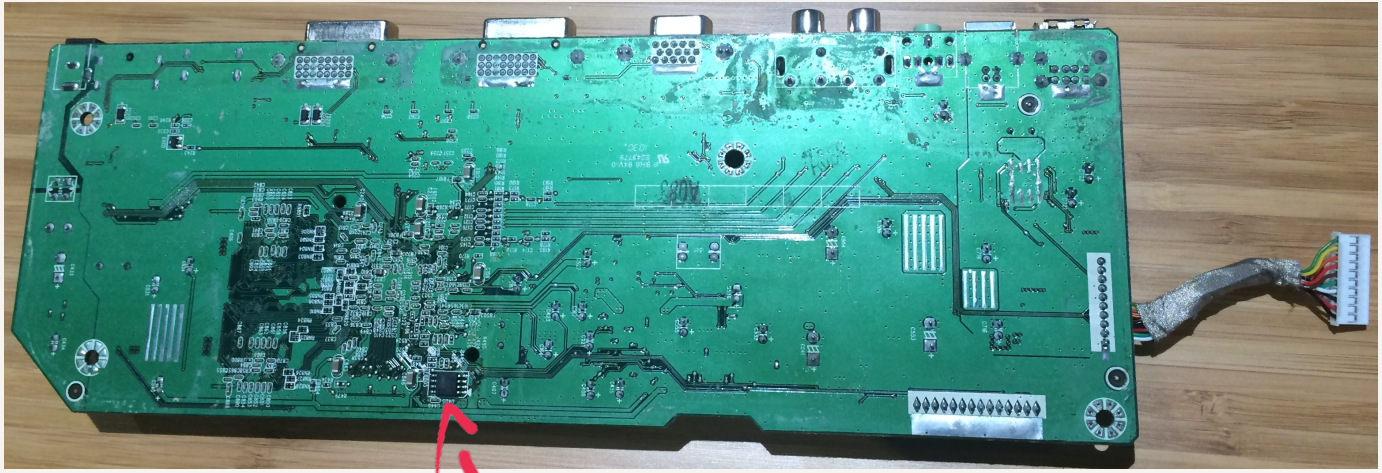
Time to void Warranty :)




saleae
1 - Ground
3 - Select
4 - Clock
5 - Data


1²C bus
?
Main MC STDP8028
I2C device 0x37
1²C bus
Max HC4053
1²C bus
select
For Devices
GP10 1²C
host
USB 2649
USB hub
USB type A x2
Daughter Board
D-
D+
USB hub USB2514B
host
D-
D+
D-
D+
D+
D-
Main Board
USB type B
to computer
For Devices
USB type A x2

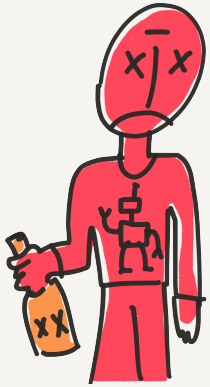

1²C bus
?
Main MC STDP8028
I2C device 0x37
1²C bus
Max HC4053
1²C bus
select
For Devices
GP10 1²C
host
USB 2649
USB hub
USB type A x2
Daughter Board
D-
D+
USB hub USB2514B
host
D-
D+
D-
D+
D+
D-
Main Board
USB type B
to computer
For Devices
USB type A x2

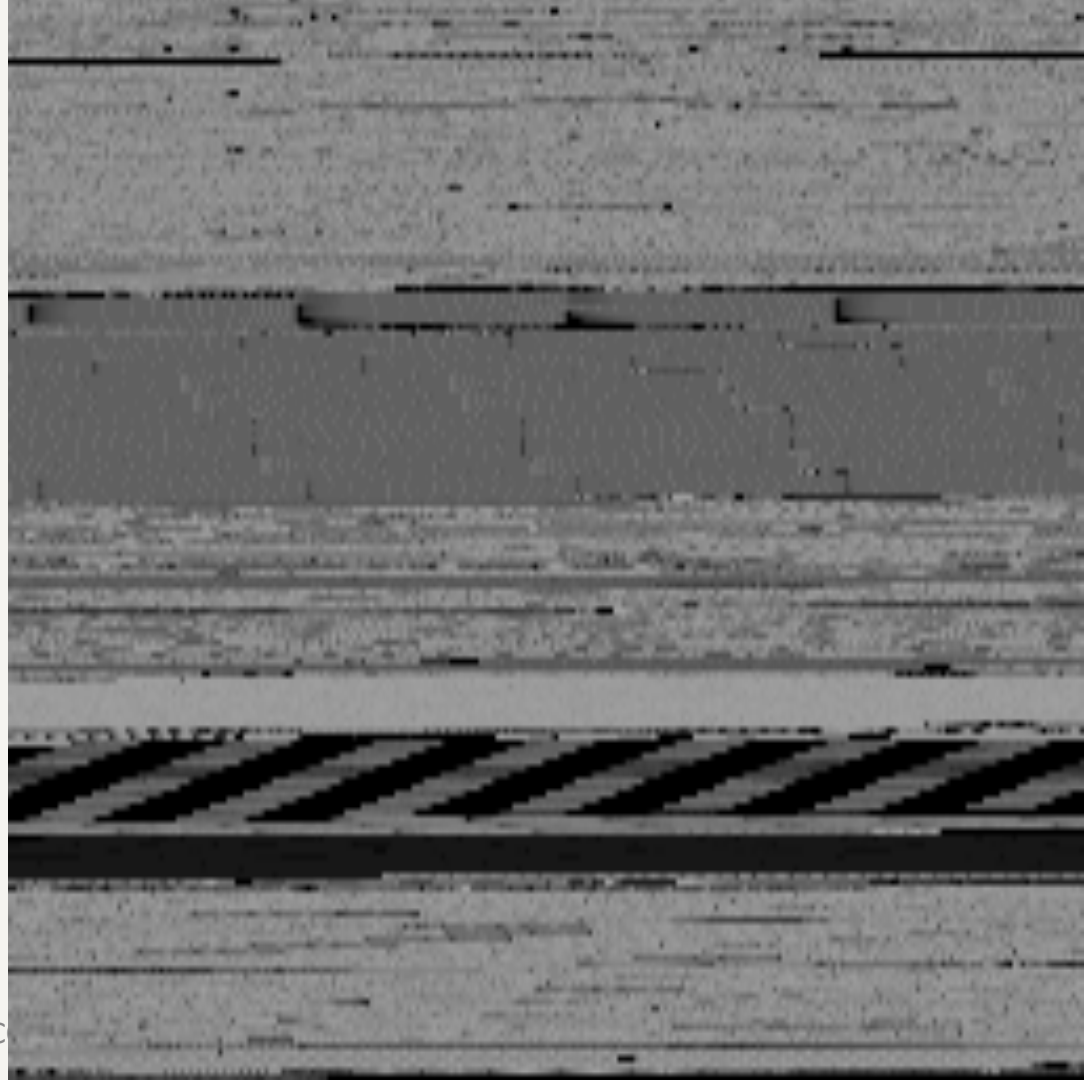


spi Flash

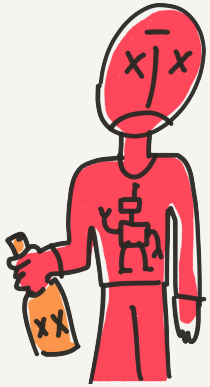


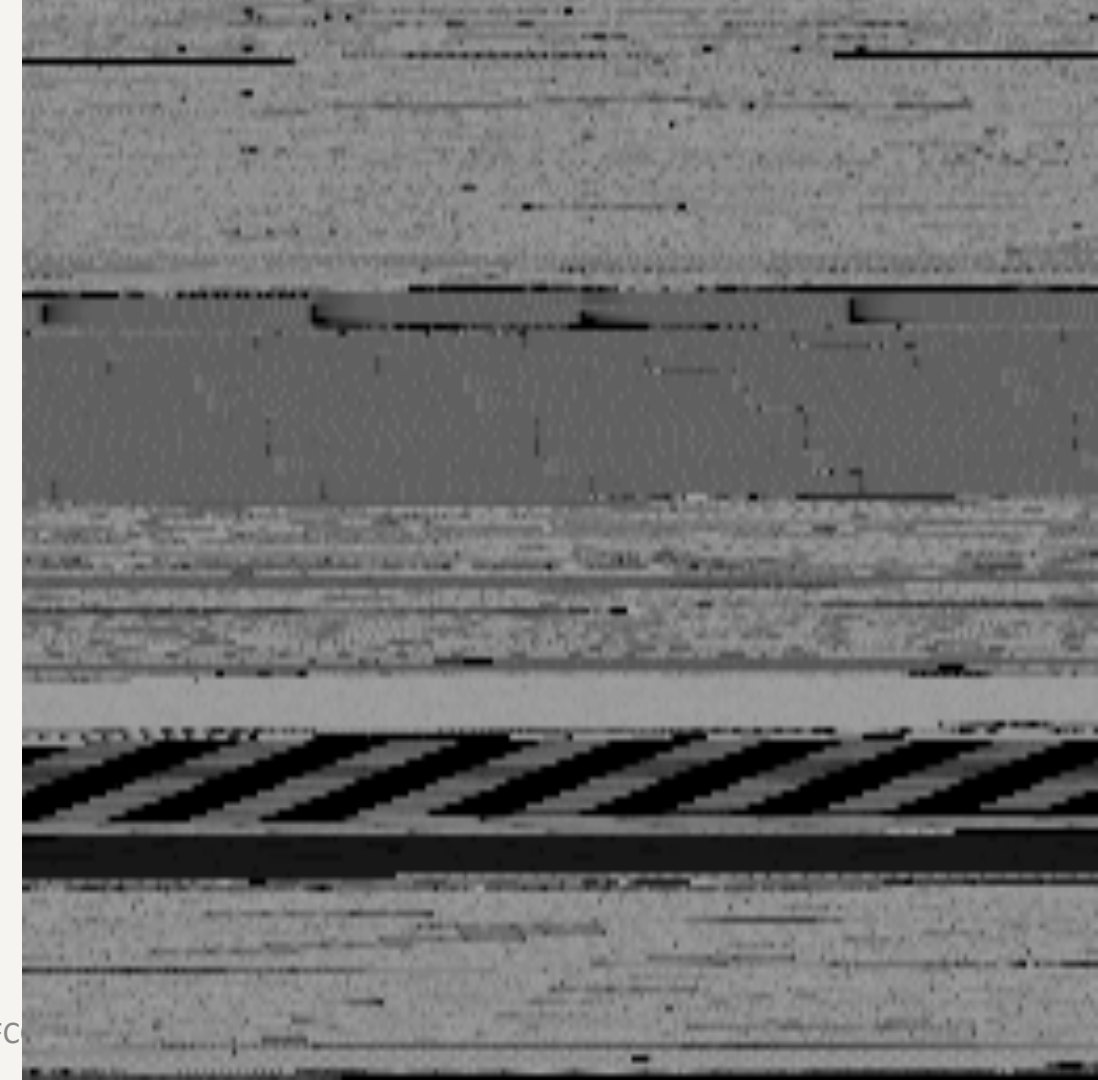
2 MB

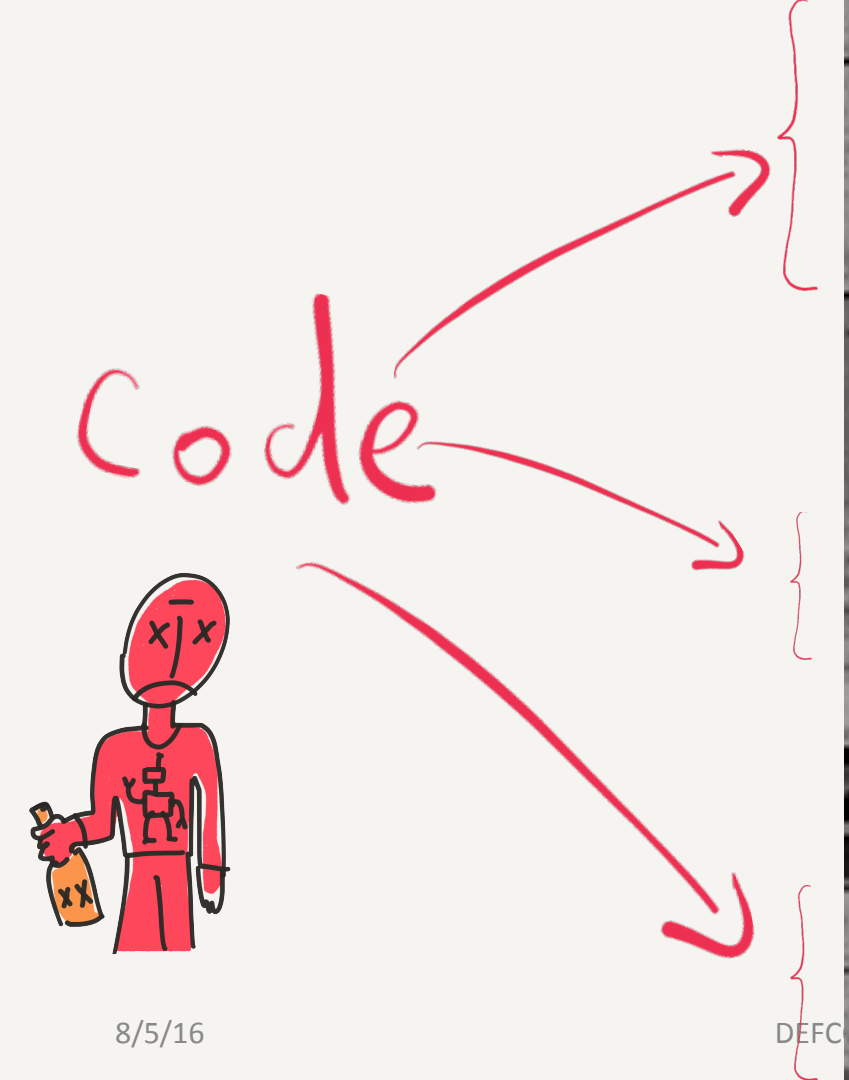


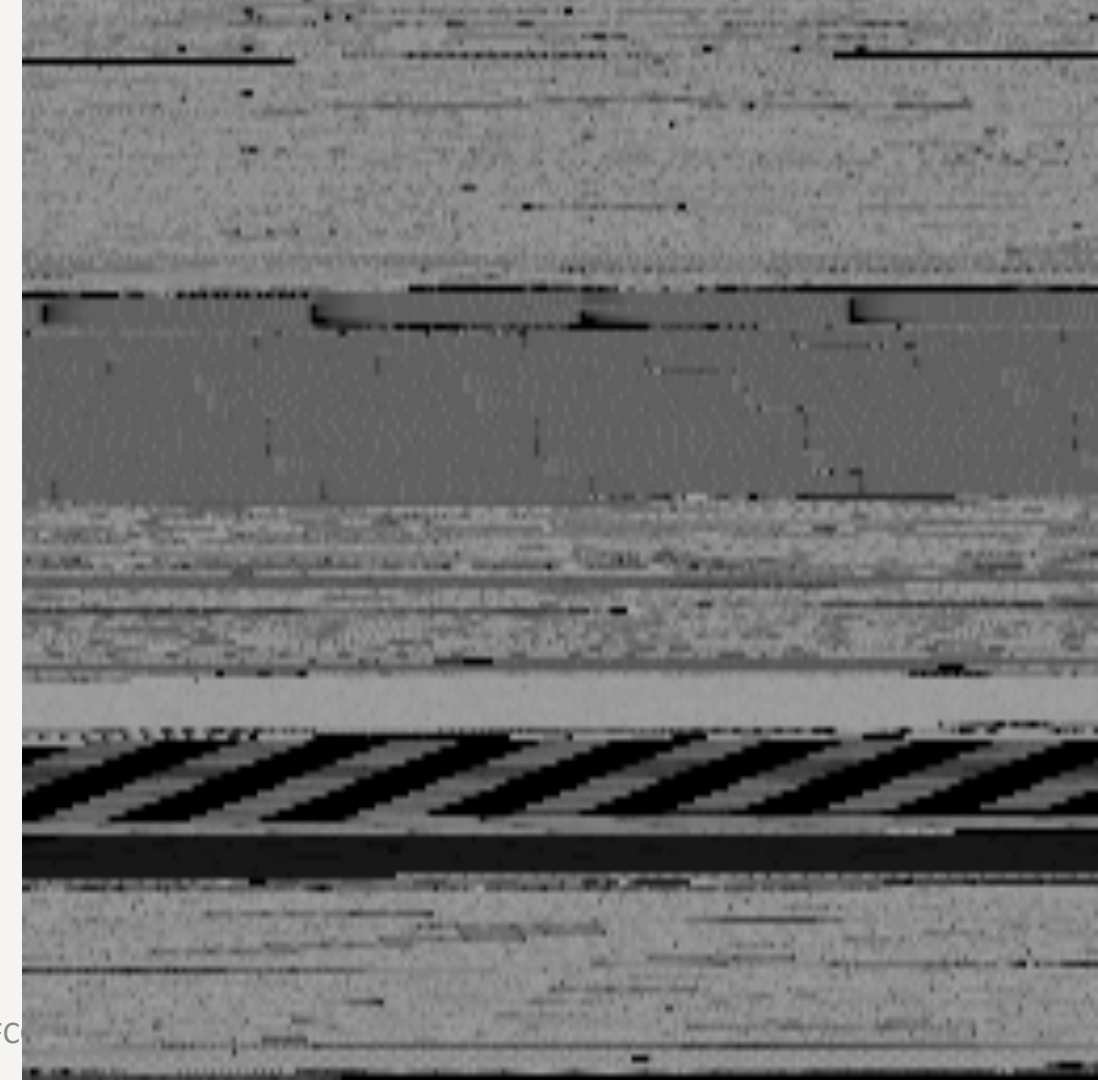


Awesome Sauce

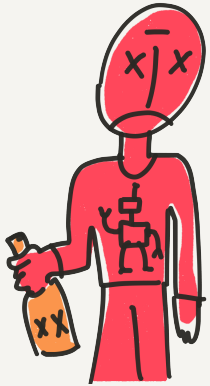


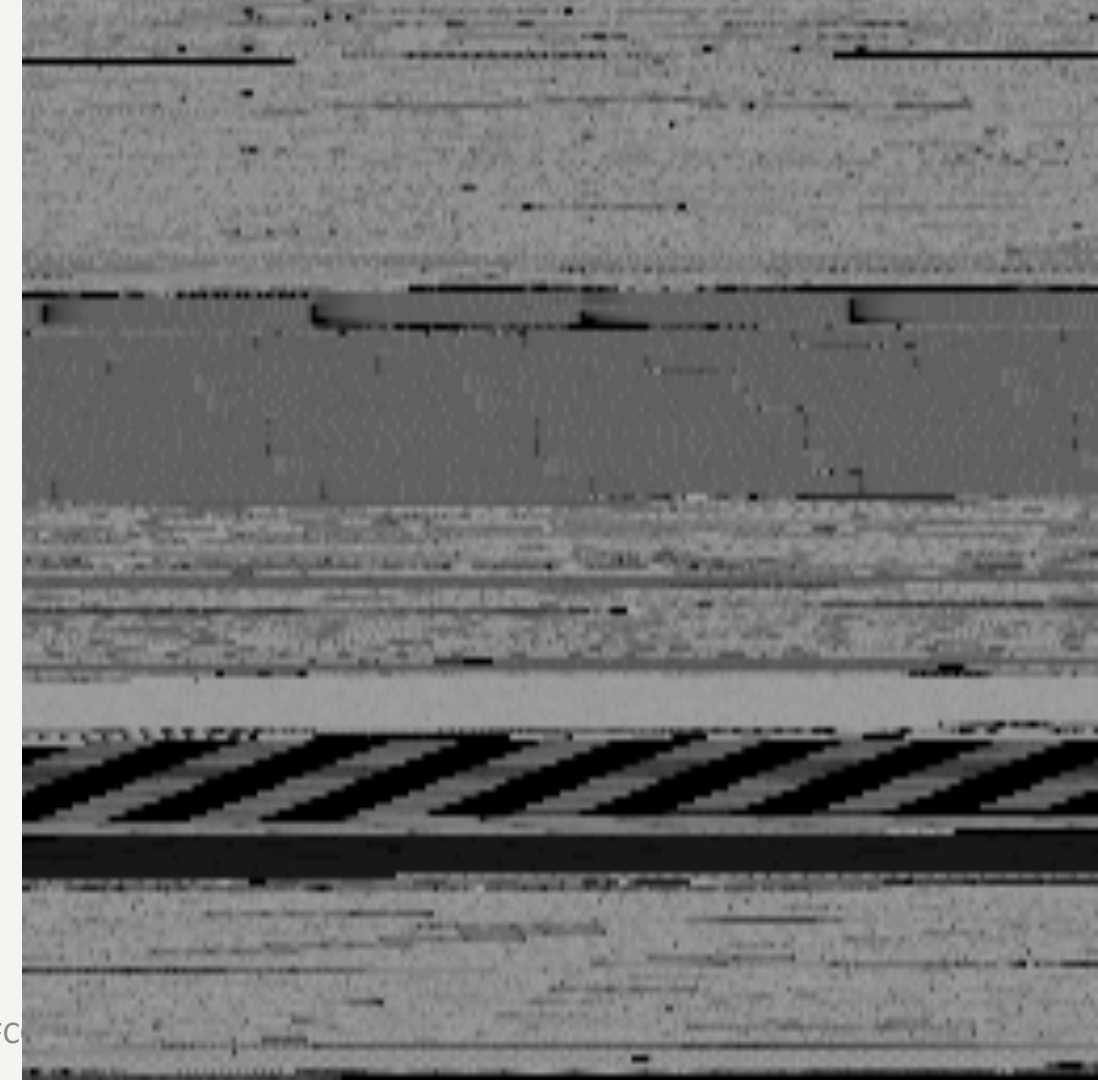



code

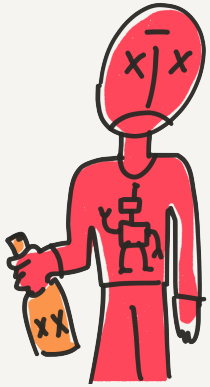


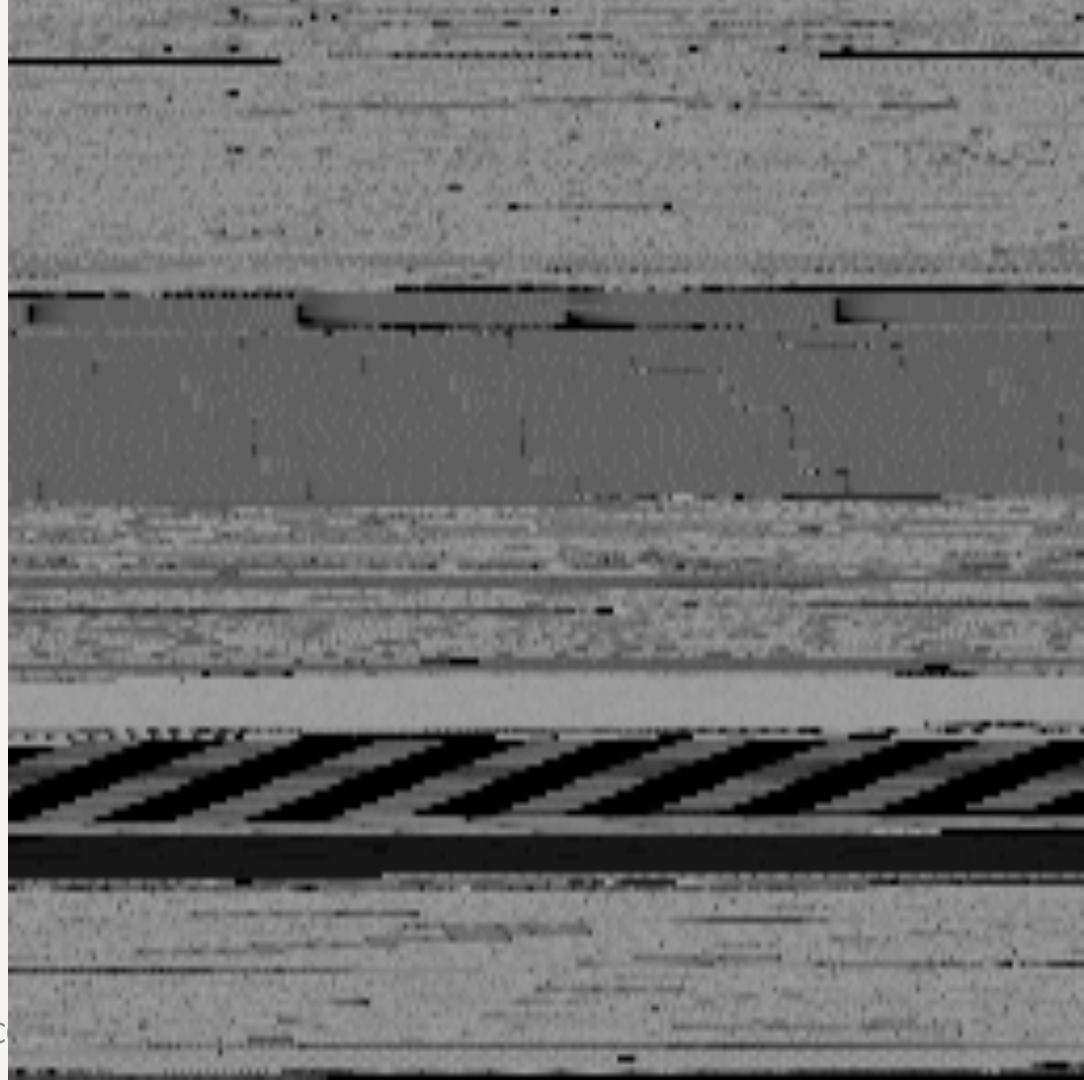
Data? ⟶ {





Compressed Data?





```
>>>>>>>>>>>>>>>>>>>>>>>>update INNOLUX
Innolux Mars Board
DELL U2410
DELL2410 Code Ver -●
------Change Input -----
------Appstest 50   0   Information ---
------Appstest 50   2   Force swtich to
------Appstest 50   3   Force swtich to
------Appstest 50   4   Force swtich to
------Appstest 50   5   Force swtich to
chsksum=0x%x
---Change Port Start---%d
Appstest 52 change source to VGA
Appstest 52 change source to DVI (HDMI2
Appstest 52 change source to DVI (HDMI3
Appstest 52 change source to DP
Appstest 52 change source to HDMI
Appstest 52 change source to COMP
```

```
---------%d
------Change Input -----
------Appstest 52   0   Information -----
------Appstest 52   1   for VGA   -----
------Appstest 52   2   for DVI1 -----
------Appstest 52   3   for DVI2 -----
------Appstest 52   4   for DP -----
------Appstest 52   5   for HDMI -----
------Appstest 52   6   for Component -----
------Appstest 52   7   for CVBS -----
------Appstest 52   8   for DDCCI Force Sleep-----
------Appstest 52   9   Factory Reset-----
------Appstest 52   10  enter Factory mode-----
appstest 57 1 :  MTO_PIPMHide
appstest 57 2 :  MTO_PIPMShow
appstest 57 3 :  MTO_NoCable
appstest 57 4 :  MTO_NoSync
appstest 57 5 :  MTO_OutOfRange
appstest 57 6 :  gm_OsdHide
appstest 57 7 :  gm_OsdShow
Appstest 72 0   ---  Display Information
Appstest 72 1 x ---  Adjust R1
Appstest 72 2 x ---  Adjust R2
Appstest 72 3 x ---  Adjust R3
Appstest 72 4 x ---  Adjust G1
```

Let's







IDA - /Users/jatin/rbs/rbs_imgw3rx/dell_mon

Library function | Data | Regular function | Unexplored | Instruction | External symbol

Functions window

Function name

- sub_86
- nullsub_2
- sub_15D7
- sub_1617
- sub_165F
- sub_16AB
- sub_1A71
- sub_1D8B
- sub_1DBE
- sub_25C5
- sub_2869
- sub_2A12
- sub_2B6B
- sub_2C30
- sub_2C73
- sub_2CCB
- sub_2D27
- sub_2D8C
- sub_2E1C
- sub_4029
- sub_43CB
- nullsub_3
- sub_4EA3
- sub_502E
- sub_5030
- sub_540A
- sub_60FF
- sub_6FB9
- sub_729A

IDA View-A | Hex View-1 | Str

```
seg000:0000 ; Input SHA256 : 5AC3E21BA2440D209B4AA2494374904C24352FA190
seg000:0000 ; Input MD5    : 0C32B7A123F52383E756FD22DD77553F
seg000:0000 ; Input CRC32  : 1C3A8743
seg000:0000
seg000:0000 ; File Name    : /Users/jatin/rbs/rbs_imgw3rx/dell_monitor/
seg000:0000 ; Format       : Binary file
seg000:0000 ; Base Address: 0000h Range: 0000h - 200000h Loaded length:
seg000:0000
seg000:0000                 .686p
seg000:0000                 .mmx
seg000:0000                 .model flat
seg000:0000
seg000:0000 ; ===========================================================
seg000:0000
seg000:0000 ; Segment type: Pure code
seg000:0000 seg000          segment byte public 'CODE' use16
seg000:0000                 assume cs:seg000
seg000:0000                 assume es:nothing, ss:nothing, ds:nothing,
seg000:0000 unk_0           db 0B8h ; ?                 ; DATA XREF: seg00
seg000:0001                 db 2 dup(0), 8Eh, 0D0h, 8Eh
seg000:0006 unk_6           db 0D8h ; ?                 ; DATA XREF: sub_A
seg000:0007                 db 8Eh, 0C0h, 0BCh, 0, 80h, 0B8h, 2Ch, 0A9
seg000:0007                                             ; DATA XREF: sub_A
seg000:0007                 db 0BEh, 2 dup(0)
seg000:0014 unk_14          db 0BFh ; ?                 ; DATA XREF: sub_A
seg000:0014                                             ; sub_A728B+103↓r
seg000:0015                 db    0
seg000:0016                 db    8
seg000:0017                 db 0B9h ; ?
seg000:0018                 db  52h ; R
seg000:0019                 db  1Bh
seg000:001A                 db  2Bh ; +
seg000:001B                 db 0CFh ; ?
seg000:001C                 db 0D1h ; ?
seg000:001D                 db 0E9h ; ?
seg000:001E                 db 0F3h ; ?
seg000:001F                 db 0A5h ; ?
seg000:0020                 db 0B8h ; ?
seg000:0021                 db    0
seg000:0022                 db    0
seg000:0023                 db  8Eh ; ?
seg000:0024                 db 0D8h ; ?
seg000:0025                 db 0FDh ; ?
seg000:0026                 dw 55B8h
seg000:0028                 db  55h ; U
seg000:0029                 db 0BFh ; ?
seg000:002A                 db    0
seg000:002B                 db  80h ; ?
seg000:002C                 db  83h ; ?
seg000:002D                 db 0EFh ; ?
seg000:002E                 db    2
seg000:002F                 db 0B9h ; ?
seg000:0030                 db    0
seg000:0031                 db    4
seg000:0032                 db 0F3h ; ?
seg000:0033                 db 0ABh ; ?
seg000:0034                 db 0FCh ; ?
seg000:0035                 db 0BDh ; ?
seg000:0036                 db    0
seg000:0037                 db    0
seg000:0038 unk_38          db 0EAh ; ?                 ; DATA XREF: seg00
```



OpenRCE.org

About
Articles
Book Store
Distributed RCE
Downloads
Event Calendar
Forums
Live Discussion
Reference Library
RSS Feeds
Search
Users
What's New

Customize Theme
blackgreen set

Flag: Tornado! Hurricane!

Forums >> IDA Pro >> Embedded Code Reversing

Topic created on: March 29, 2008 18:46 CDT by memo5.

Hello All
I'm working on reversing an embedded system, its my first attempt and need some advice, first of all the program file is in Intel-Hex format, so the code address is obtained by IDA, the target platform is x86 compatible, I'm facing some difficulties in setting the segments addresses and values cs,ds,es segments values.
I appreciate any help.

igorsk — March 31, 2008 07:54.27 CDT

What's the CPU? Is the code 16-bit or 32-bit? Is it the initial code (bootloader, BIOS) or is it something that's more high level? Are there datasheets available? Is the binary available for download somewhere or can you upload it?
In short, list as much info as you can and try to describe the actual problem you're having in more details. "Some difficulties" is not much to work with.
See also this.

memo5 — April 11, 2008 21:10.13 CDT

Ok igorsk thank you for your advice.
And hope that this information will help.

The CPU is Turbo186 the code is 16 bit.
The CPU run in extended mode using 24bit addressing capability.
-The paragraph is not 16 byte its 256 byte so the CPU address space is 16MB-
and the EA calculated as: EA = (segment << 8) + offset.

The code is not just a Bootloader or BIOS it's the full firmware of a multimedia related control board.
Unfortunately I can't upload the binary file, and it's not published any ware.
The firmware mapped to address 0x600000, and the size of the firmware is 2MB

When I load the binary into IDA and disassemble the code I got code like this:

```
seg000:6005CD  push   bp
seg000:6005CE  mov    bp, sp
seg000:6005D0  push   si
seg000:6005D1  mov    si, [bp+6]
seg000:6005D4  push   si
seg000:6005D5  call   far ptr 6019h:0B1Fh          --> Call 601900:0B1F
seg000:6005DA  pop    cx
seg000:6005DB  cmp    ax, 1
seg000:6005DE  jnz    short near ptr 5E4h       ---+
seg000:6005E0  xor    ax, ax                      |
seg000:6005E2  jmp    short near ptr 5EBh       ---+   |
seg000:6005E4  ------------------------------     |   |
seg000:6005E4  push   si                        | <--+
seg000:6005E5  call   far ptr 6306h:1Ch         |      --> Call 630600:1C
seg000:6005EA  pop    cx                        |
seg000:6005EB  pop    si                  <---+
seg000:6005EC  pop    bp
seg000:6005ED  retf
```

You can notice that IDA could not recognize calls or jumps to near or far addresses.

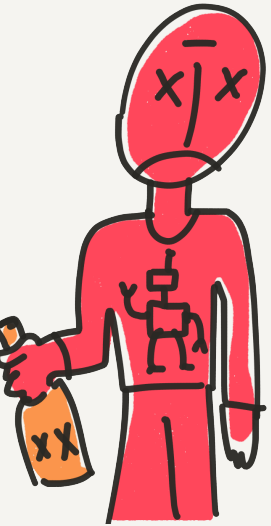
To be specific how can I make IDA recognize the (24Bit) addressing mode, and correctly identify calls and jumps to other addresses?

igorsk — April 12, 2008 20:58.17 CDT

I'm afraid it's not possible with current IDA version. The PC processor module is hardcoded to the standard x86 segmentation (ea = segment<<4 + offset). The best you can do is report this issue to Ilfak. This will probably need a new configuration flag in the processor module.
In the meantime you could probably write a plugin or an IDC script which would walk through all the call instructions and fix the cross references.

#!*¥!# x86

This game sucks...



IDA - /Users/jatin/rbs/rbs_imgw3rx/dell_mon

Library function | Data | Regular function | Unexplored | Instruction | External symbol

Functions window

Function name

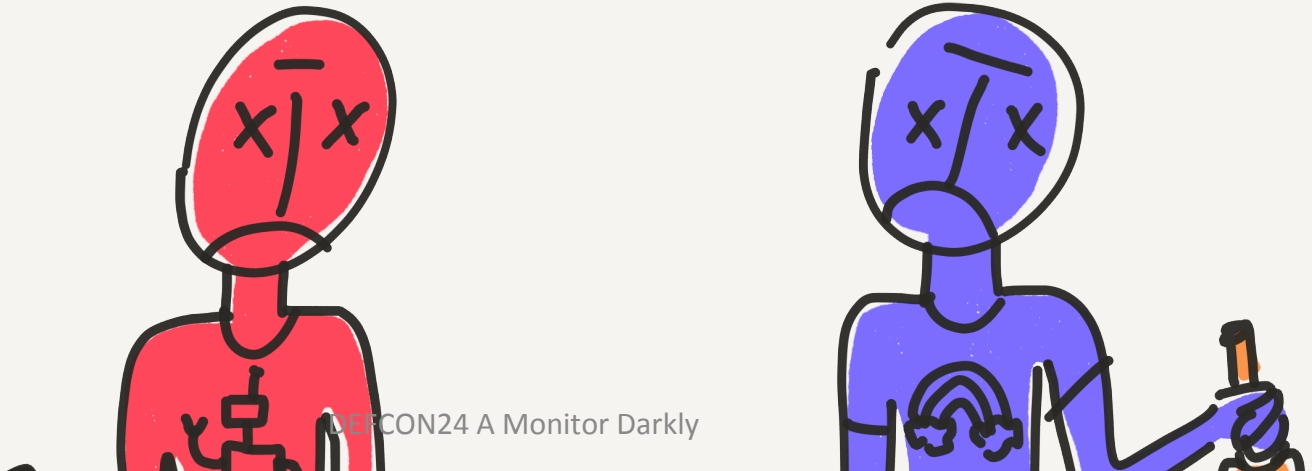
- sub_86
- nullsub_2
- sub_15D7
- sub_1617
- sub_165F
- sub_16AB
- sub_1A71
- sub_1D8B
- sub_1DBE
- sub_25C5
- sub_2869
- sub_2A12
- sub_2B6B
- sub_2C30
- sub_2C73
- sub_2CCB
- sub_2D27
- sub_2D8C
- sub_2E1C
- sub_4029
- sub_43CB
- nullsub_3
- sub_4EA3
- sub_502E
- sub_5030
- sub_540A
- sub_60FF
- sub_6FB9
- sub_729A

IDA View-A | Hex View-1 | Str

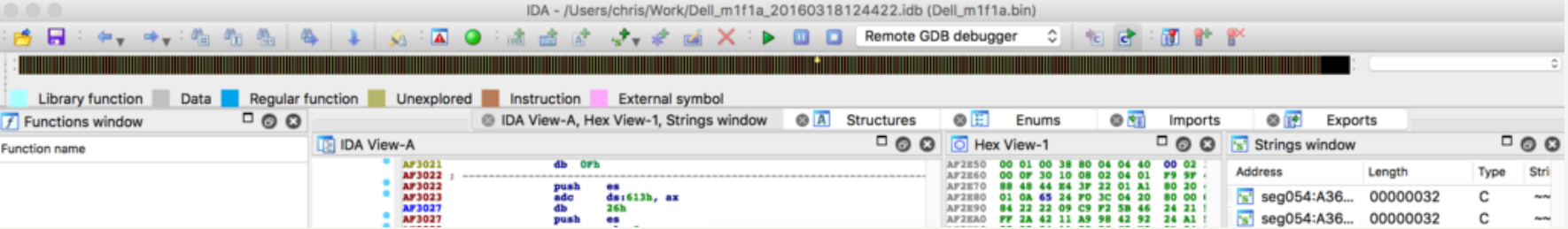
```
seg000:0000 ; Input SHA256 : 5AC3E21BA2440D209B4AA2494374904C24352FA190
seg000:0000 ; Input MD5    : 0C32B7A123F52383E756FD22DD77553F
seg000:0000 ; Input CRC32  : 1C3A8743
seg000:0000
seg000:0000 ; File Name   : /Users/jatin/rbs/rbs_imgw3rx/dell_monitor/
seg000:0000 ; Format      : Binary file
seg000:0000 ; Base Address: 0000h Range: 0000h - 200000h Loaded length:
seg000:0000
seg000:0000                 .686p
seg000:0000                 .mmx
seg000:0000                 .model flat
seg000:0000
seg000:0000 ; ===========================================================
seg000:0000
seg000:0000 ; Segment type: Pure code
seg000:0000 seg000          segment byte public 'CODE' use16
seg000:0000                 assume cs:seg000
seg000:0000                 assume es:nothing, ss:nothing, ds:nothing,
seg000:0000 unk_0           db 0B8h ; ?                 ; DATA XREF: seg00
seg000:0001                 db 2 dup(0), 8Eh, 0D0h, 8Eh
seg000:0006 unk_6           db 0D8h ; ?                 ; DATA XREF: sub_A
seg000:0007                 db 8Eh, 0C0h, 0BCh, 0, 80h, 0B8h, 2Ch, 0A9
seg000:0007                                             ; DATA XREF: sub_A
seg000:0007                 db 0BEh, 2 dup(0)
seg000:0014 unk_14          db 0BFh ; ?                 ; DATA XREF: sub_A
seg000:0014                                             ; sub_A728B+103↓r
seg000:0015                 db    0
seg000:0016                 db    8
seg000:0017                 db 0B9h ; ?
seg000:0018                 db  52h ; R
seg000:0019                 db  1Bh
seg000:001A                 db  2Bh ; +
seg000:001B                 db 0CFh ; ?
seg000:001C                 db 0D1h ; ?
seg000:001D                 db 0E9h ; ?
seg000:001E                 db 0F3h ; ?
seg000:001F                 db 0A5h ; ?
seg000:0020                 db 0B8h ; ?
seg000:0021                 db    0
seg000:0022                 db    0
seg000:0023                 db  8Eh ; ?
seg000:0024                 db 0D8h ; ?
seg000:0025                 db 0FDh ; ?
seg000:0026                 dw 55B8h
seg000:0028                 db  55h ; U
seg000:0029                 db 0BFh ; ?
seg000:002A                 db    0
seg000:002B                 db  80h ; ?
seg000:002C                 db  83h ; ?
seg000:002D                 db 0EFh ; ?
seg000:002E                 db    2
seg000:002F                 db 0B9h ; ?
seg000:0030                 db    0
seg000:0031                 db    4
seg000:0032                 db 0F3h ; ?
seg000:0033                 db 0ABh ; ?
seg000:0034                 db 0FCh ; ?
seg000:0035                 db 0BDh ; ?
seg000:0036                 db    0
seg000:0037                 db    0
seg000:0038 unk_38          db 0EAh ; ?                 ; DATA XREF: seg00
```

Computers are hard.

Let's ask Ilfak

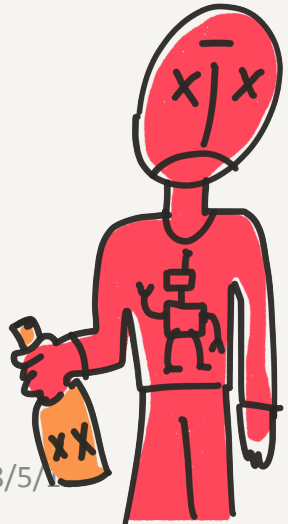



long insightful
Explanation of
IDA + Turbo 186
and here is
mars.hex.idb



Segment City

& fail



Execute script

Snippet list

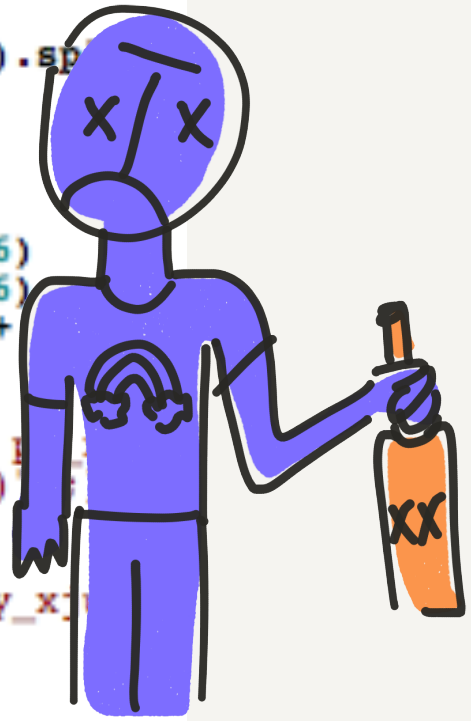
Name

ResetAnalysis

Default snippet

Please enter script body

```
import idaapi
def xjump():
    ea = ScreenEA()
    da_call = GetOpnd(ea, 0).sp
    for c in da_call:
        if ":" in c:
            op = c
    op = op.split(':')
    seg = int(op[0][:-1], 16)
    off = int(op[1][:-1], 16)
    new_ea = (seg * 0x100) +
    Jump(new_ea)

idaapi.CompileLine('static
RunPythonStatement("xjump()

AddHotkey('Ctrl+Space', 'py_x
```



RegRead

Request			
Name	**Length**	**Value**	**Comment**
Length	1	Length of message + 1	
Command	1	0x06	
Register Address	2		
Response			
Length	1	Length of message + 1	
Command	1	0x06	
Register Address	2		
Register Value	2		

RunCode

Request			
Name	**Length**	**Value**	**Comment**
Length	1	Length of message + 1	
Command	1	0x1D	
Address	2		
Response			
ACK/NACK			

RAM WRITE

Request

NAME	LEN	VALUE
Length	1	Message LEN 1
CMD	1	0 x 13
REG Address	2	

Response

Length	1	Message LEN 1
CMD	1	0 x 13
REG Address	2	
REG VAL	2	

hijack Appfest....

(our first helloworld)



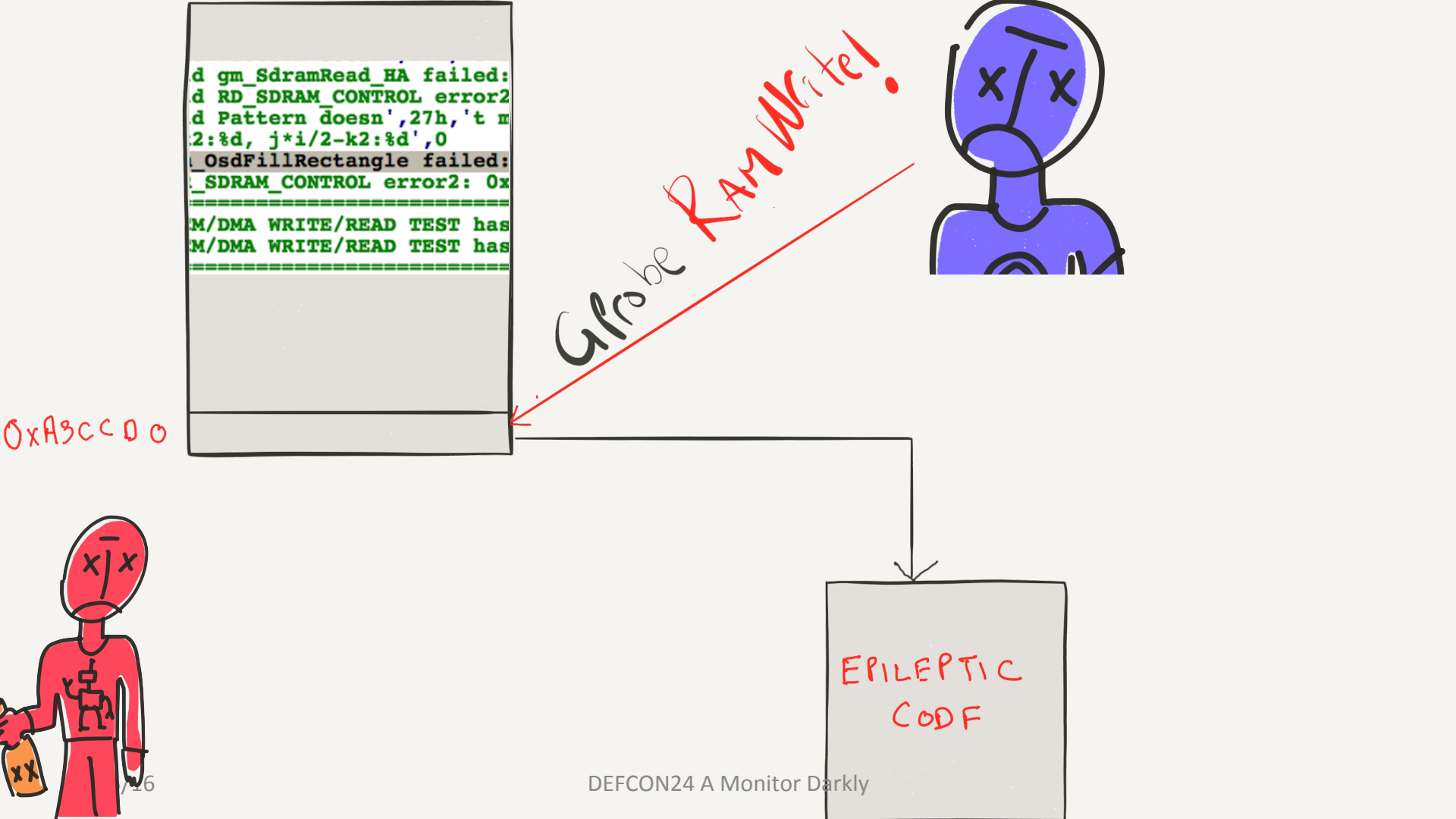
```
d gm_SdramRead_HA failed:
d RD_SDRAM_CONTROL error2
d Pattern doesn',27h,'t m
2:%d, j*i/2-k2:%d',0
_OsdFillRectangle failed:
_SDRAM_CONTROL error2: 0x
==========================
M/DMA WRITE/READ TEST has
M/DMA WRITE/READ TEST has
==========================
```

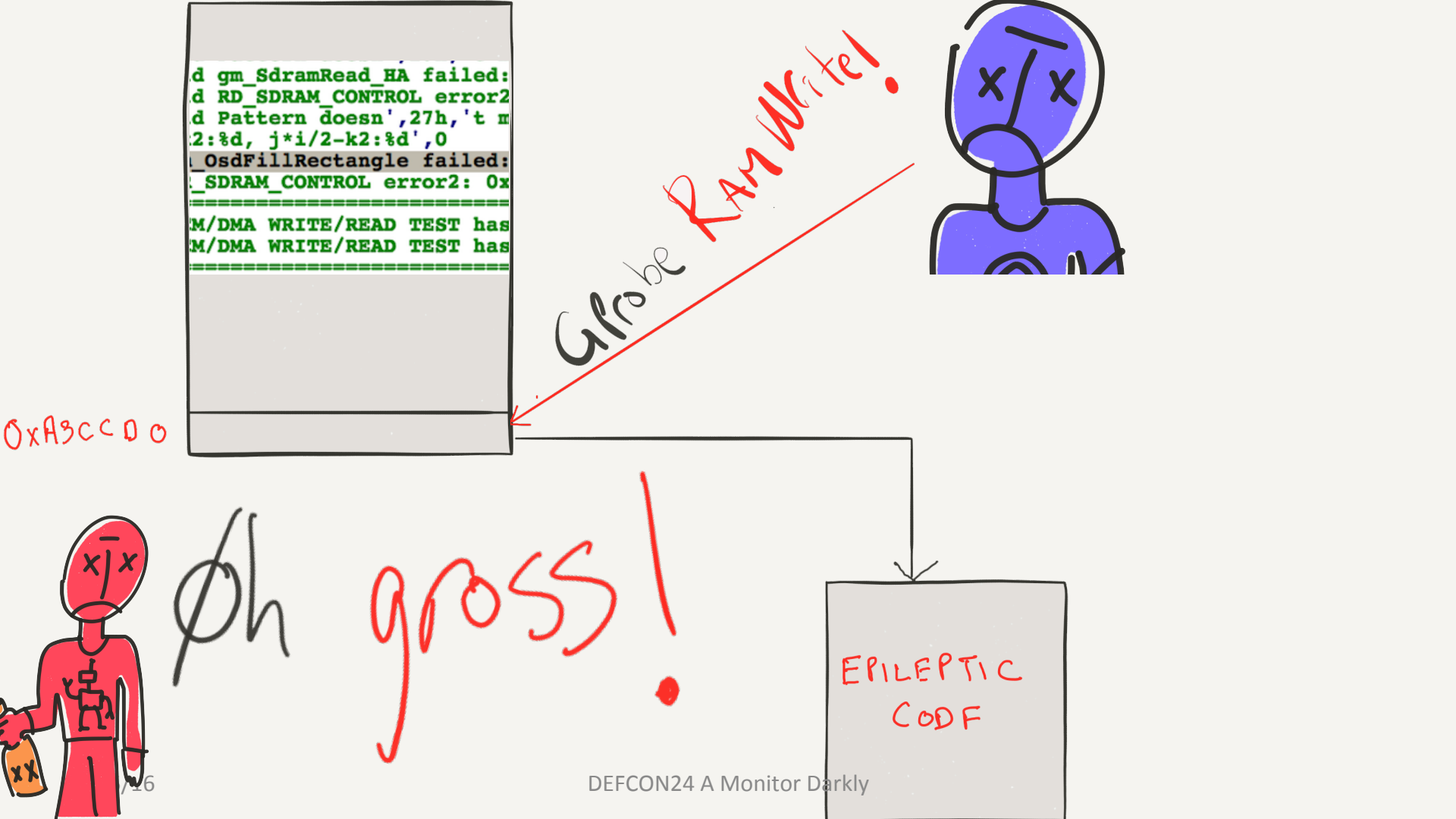
hijack Apptest....
(our first helloworld)

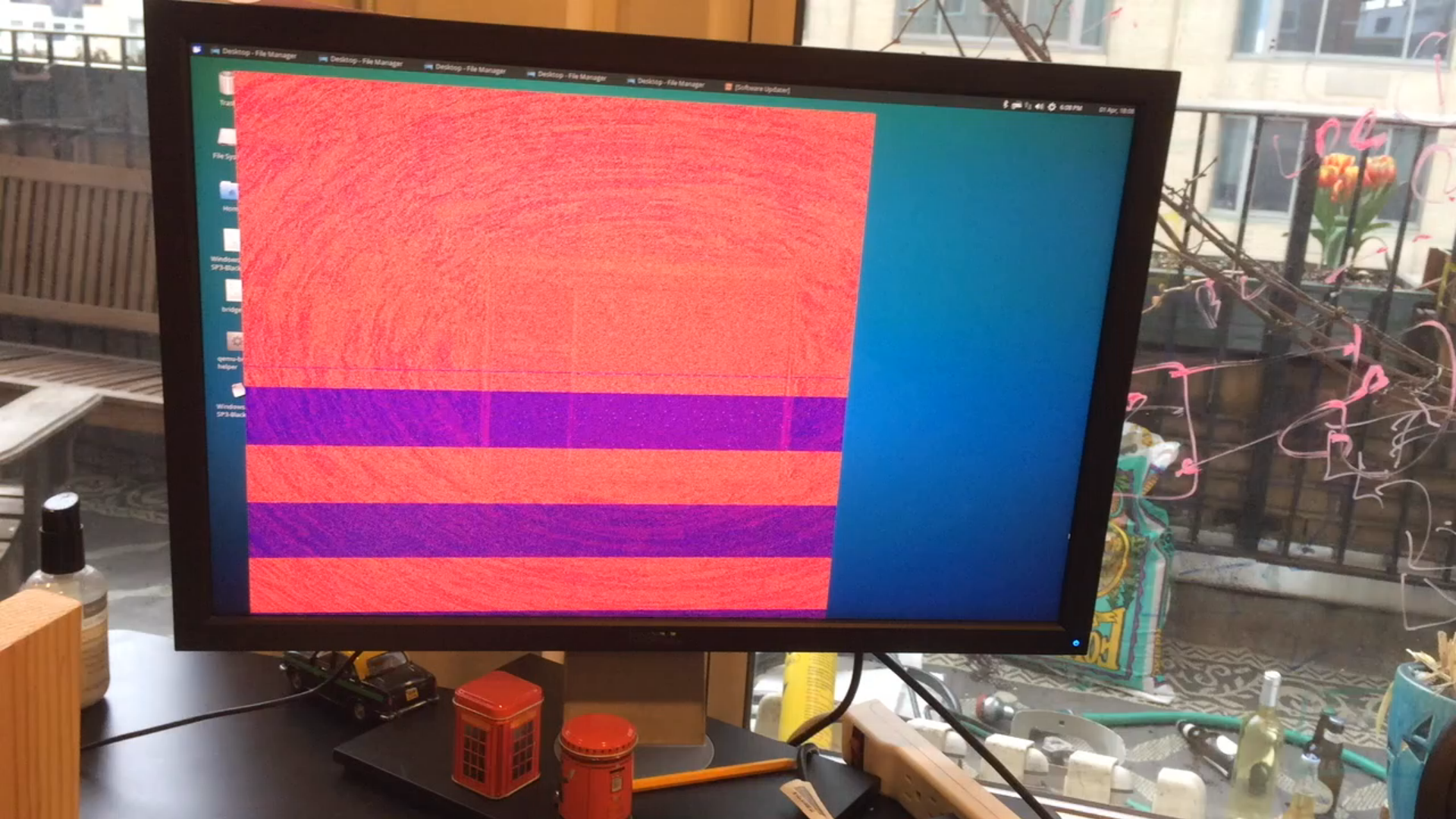
0xA3CCD0



EPILEPTIC CODE


d gm_SdramRead_HA failed:
d RD_SDRAM_CONTROL error2
d Pattern doesn',27h,'t m
2:%d, j*i/2-k2:%d',0
_OsdFillRectangle failed:
_SDRAM_CONTROL error2: 0x
M/DMA WRITE/READ TEST has
M/DMA WRITE/READ TEST has
Globe RAM Write!
0xA3CCD0
EPILEPTIC CODF


d gm_SdramRead_HA failed:
d RD_SDRAM_CONTROL error2
d Pattern doesn',27h,'t m
2:%d, j*i/2-k2:%d',0
_OsdFillRectangle failed:
_SDRAM_CONTROL error2: 0x
M/DMA WRITE/READ TEST has
M/DMA WRITE/READ TEST has
Globe RamWrite!
0xA5CCD0
Oh gross!
EPILEPTIC CODE


Desktop - File Manager
Desktop - File Manager
Desktop - File Manager
Desktop - File Manager
Desktop - File Manager
[Software Updater]

Monitor Memory world

"OSD Firmware"

0xA0000 ——————— 0xB0000

```
:A7FD33                 xor     ax, ax
:A7FD35                 push    ax
:A7FD36                 push    cs
:A7FD37                 mov     ax, 91h ; '?'
:A7FD3A                 push    ax
:A7FD3B                 call    far ptr 0F000h:252h ; print function!
:A7FD40                 add     sp, 6
:A7FD43                 xor     ax, ax
:A7FD45                 push    ax
:A7FD46                 push    cs
:A7FD47                 mov     ax, 0A3h ; '?'
:A7FD4A                 push    ax
:A7FD4B                 call    far ptr 0F000h:252h
:A7FD50                 add     sp, 6
:A7FD53                 xor     ax, ax
:A7FD55                 push    ax
```

Monitor Memory world

"OSD Firmware"

???????

0xA0000 ——————— 0xB0000

0xF0000

```
:A7FD33                xor     ax, ax
:A7FD35                push    ax
:A7FD36                push    cs
:A7FD37                mov     ax, 91h ; '?'
:A7FD3A                push    ax
:A7FD3B                call    far ptr 0F000h:252h ; print function!
:A7FD40                add     sp, 6
:A7FD43                xor     ax, ax
:A7FD45                push    ax
:A7FD46                push    cs
:A7FD47                mov     ax, 0A3h ; '?'
:A7FD4A                push    ax
:A7FD4B                call    far ptr 0F000h:252h
:A7FD50                add     sp, 6
:A7FD53                xor     ax, ax
:A7FD55                push    ax
```

Monitor Memory World

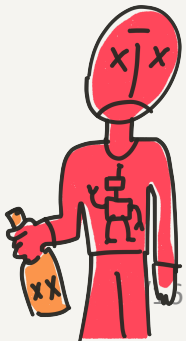
"OSD Firmware"

0x0000 ——— 0xB000

0xF000

???????

Let's dump!



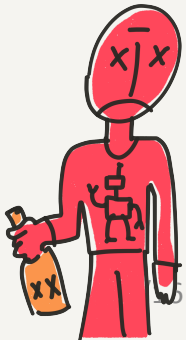
Monitor Memory world

"OSD Firmware"

0x0000 ——— 0xB000

0xF000

???????



mkay usb dump



Monitor Memory World

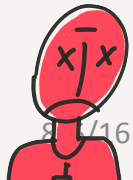
"OSD Firmware"

0xA0000 ——————— 0xB0000

???????

0xF0000

Dump...





Monitor Memory world

"OSD Firmware"

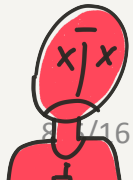
0x0000 ——— 0x8000

??????

0xF000

Dump...

Dump...





Monitor Memory World

"OSD Firmware"

0x A000000 ——————— 0x B000000

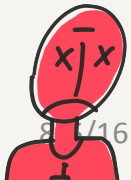
???????

0x F000000

Dump...

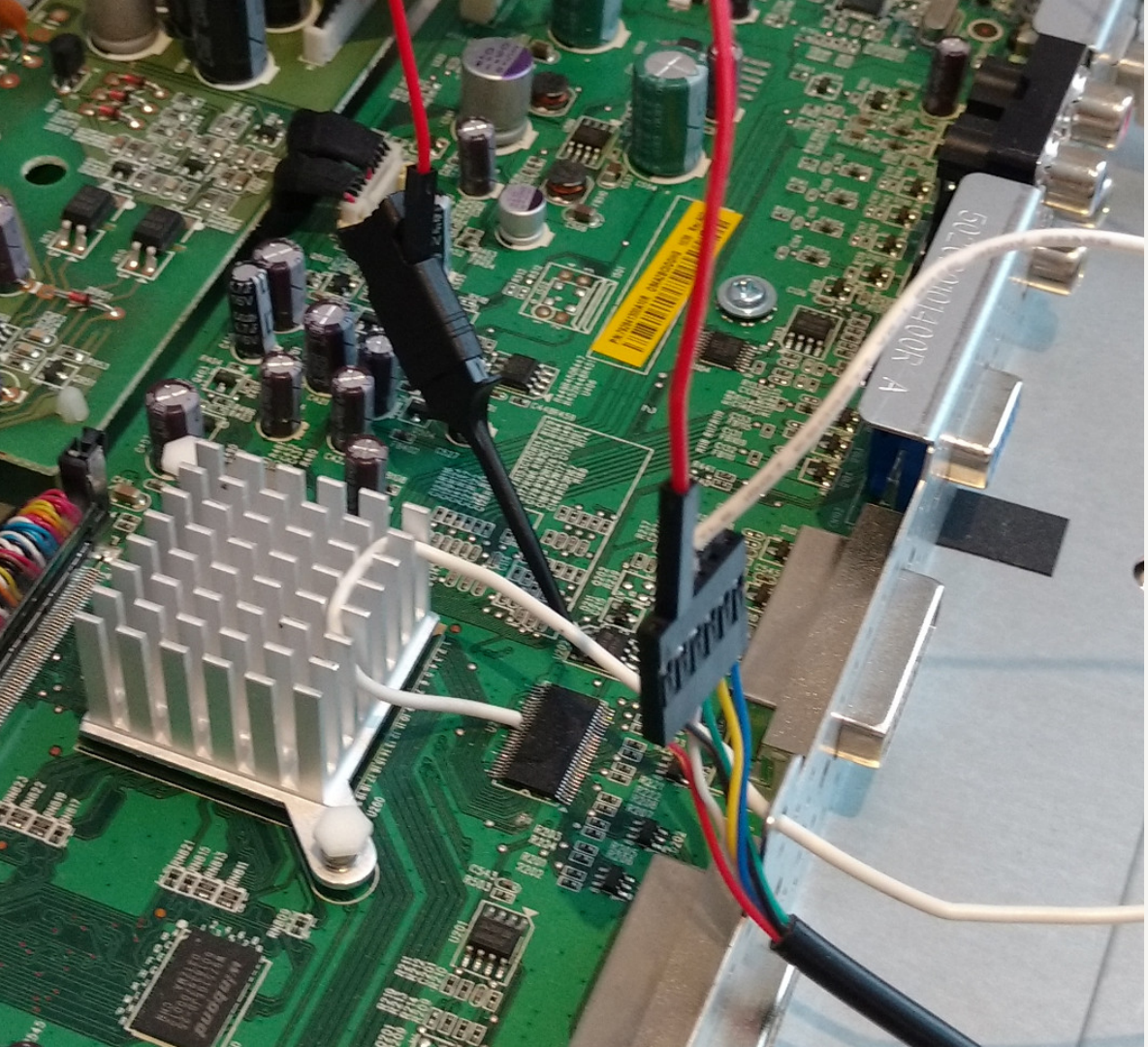
Dump...

Dump...






You Dumb Too Slow =(





```
Welcome to minicom 2.7

OPTIONS: I18n
Compiled on Sep  6 2015, 19:49:19.
Port /dev/ttyUSB0, 17:29:46

Press CTRL-A Z for help on special keys

~~~~Exit MesInvState(Fly)~~~~
~~~~Enter MesInvState(Fly)~~~~
~~~~Exit MesInvState(Fly)~~~~
~~~~Enter DefaultState~~~~
~~~~Exit DefaultState~~~~
~~~~Enter MesInvState(Fly)~~~~
~~~~Exit MesInvState(Fly)~~~~
~~~~Enter MesInvState(Fly)~~~~
~~~~Exit MesInvState(Fly)~~~~
~~~~Enter ValidMode_StateState~~~~
~~~~Exit ValidMode_StateState~~~~
~~~~Enter MesInvState(Fly)~~~~
```



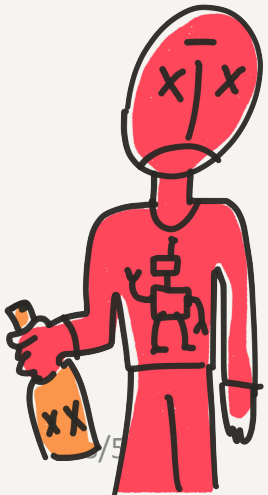
Monitor Memory world

"OSD Firmware"

0xA0000 ——— 0xB0000

0xF0000

???????



OCM Executable
(On-chip microcontroller)

IROM
SoC Driver



Like a million years of googling....

(and clicking on all the links)

VMware Fusion File Edit View Virtual Machine Window Help

SecOps

Google Chrome

STDP8028-AB_Regis ×

www.doc88.com/p-2973089571270.html

下载 收藏 20 / 705

		1 = TNR3 processing is disabled on left half of display output image. Normal TNR3 operation on right side of display. (Done by forcing motion value to max motion tnr coefficients)
15:10	Reserved	Reserved

Reserved Register: 0xC818

0xC81A	DP_RCAL_RESULT	RO
POD: 0x0000		
Display Port calibration Result		

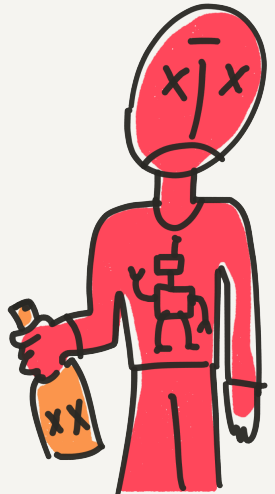
BIT	BIT NAME	FUNCTION
3:0	DPRX_RCAL_RESULT	DPRX Link Calibration Result
7:4	Reserved	Reserved
15:8	Reserved	Reserved

3.3.3 Display Signal Adjustment for TTL Panel Signals and Output Clocks

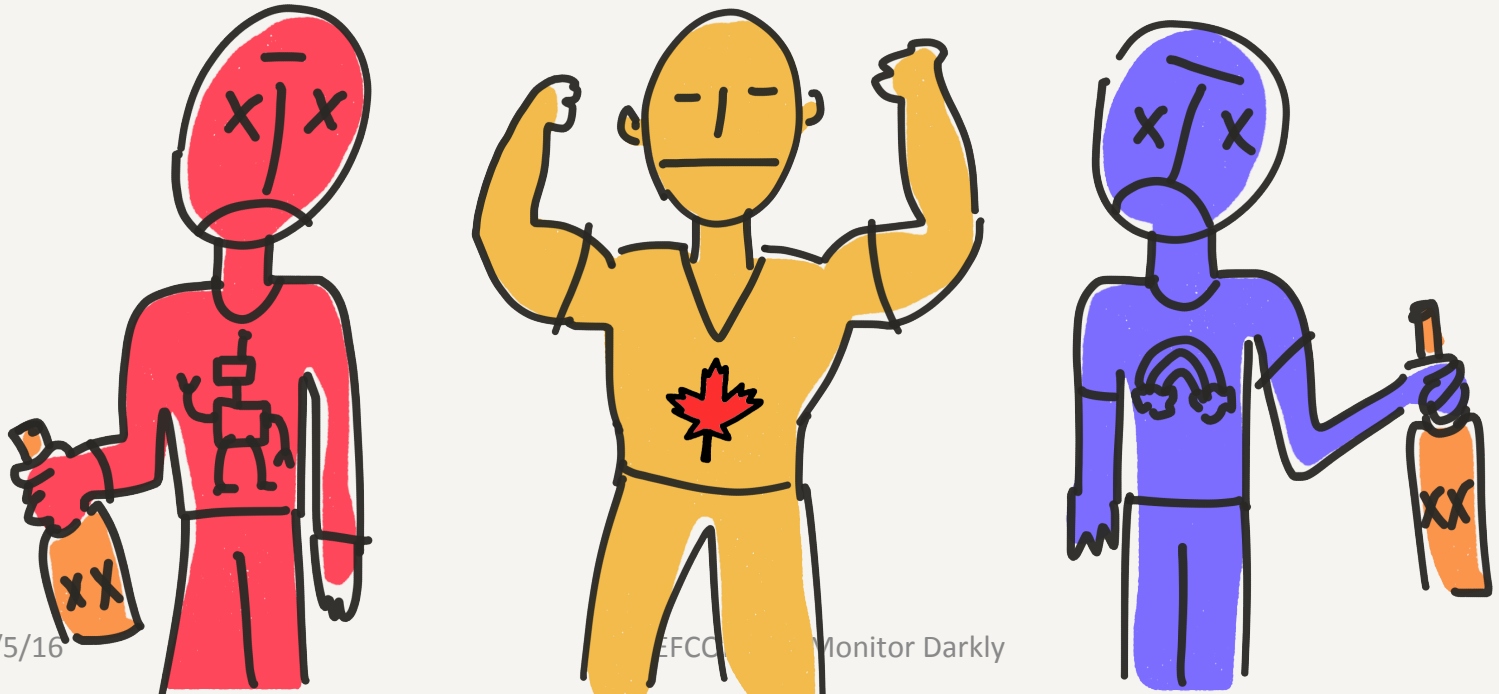
0xC81C	TIMING_CONFIG	RW
POD: 0x0000		
Timing Configuration for Display and Audio Outputs		

BIT	BIT NAME	FUNCTION
0	DCLK_INV	Invert DCLK pin
1	DCLK_DIS	Disable DCLK output pin
2	DCLK_RATE_INV	The transmitted display clock frequency at the oDCLK pad is 1x the internal pixel rate for single wide TTL output, 1/2x the internal pixel rate for double wide TTL output, and 1/2x the

OPEN IN A VM →



Let's Display Picture!





where to transfer image

1 where to transfer image

2 how to display transfered image

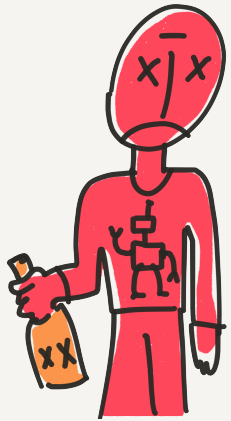
1 where to transfer image

2 how to display transfered image

3 ... what is a color



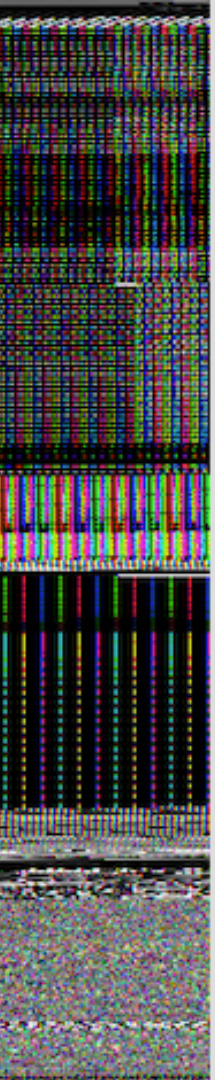
must be In The Firmware

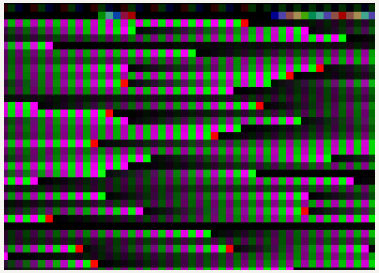


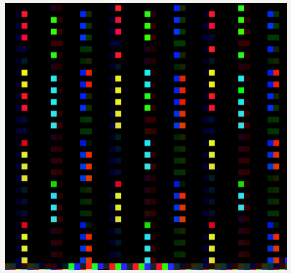
Drunken "Discussion"

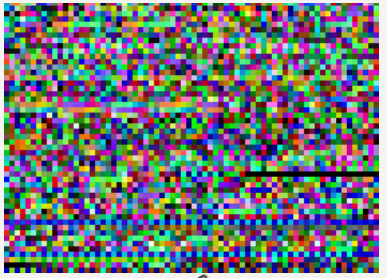


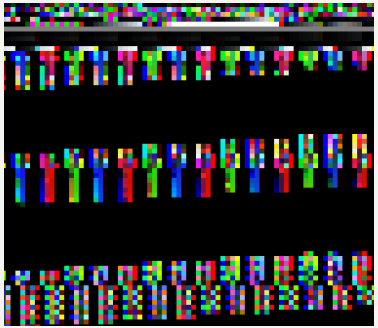




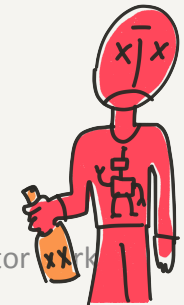




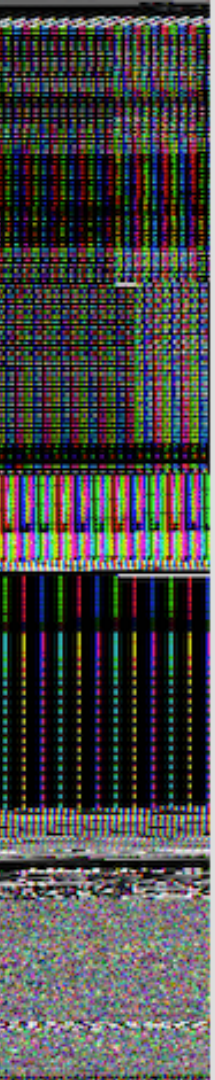


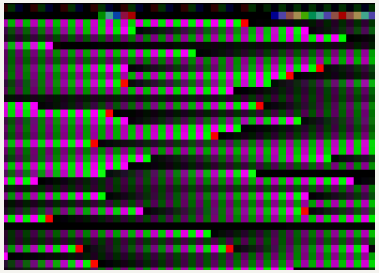


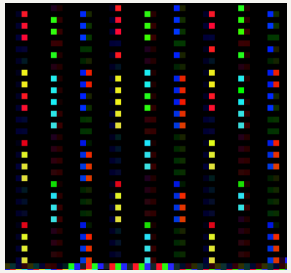
Not the Dell Logo!

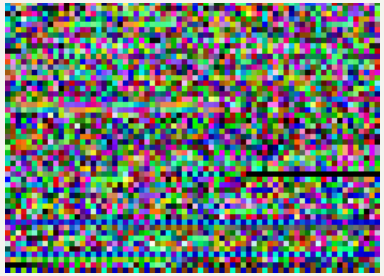


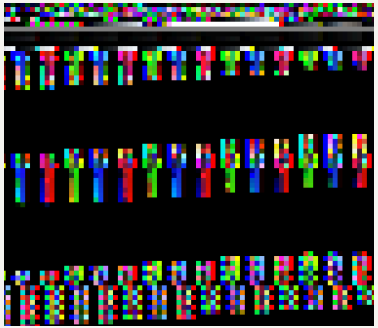




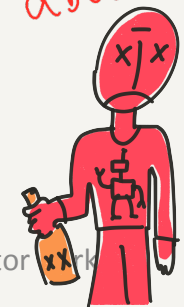


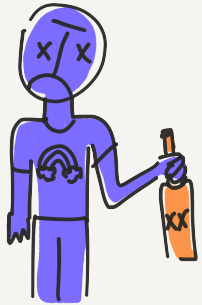


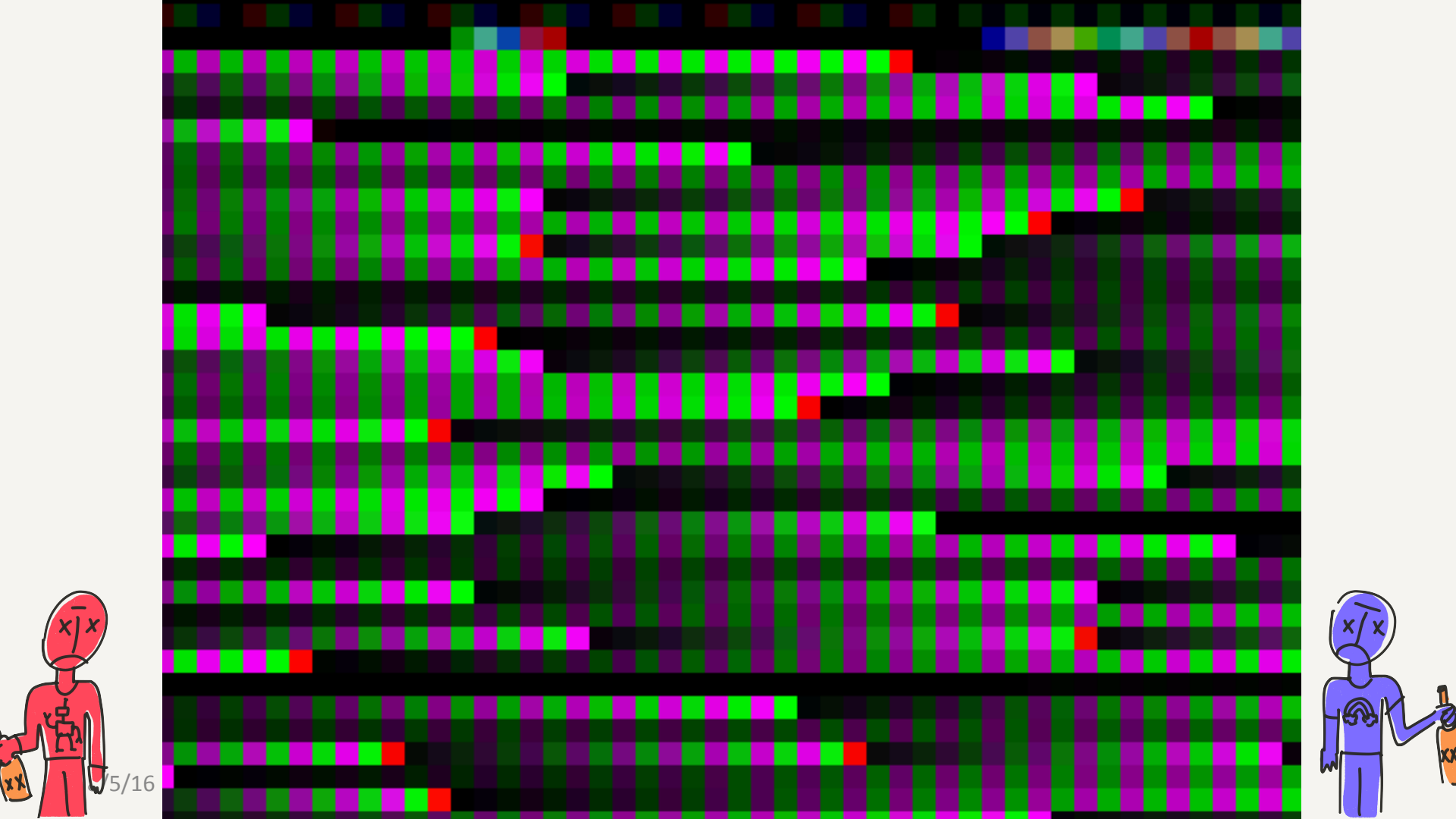




But what about this?







Dump Dump Dump



```
E9 8F 2E C1 5D F7 11 5B 28 1C F7 CD B1 DB E5 96 0F 35 AE A9 D5 B0 50 DD FD 96 F3 6B 00 7C 6F C0 FF 92 ED 2A 7C 76 FF 43 87 47 83
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01 AB 03 16 00 18 00 00 00 E8 00 CB 03 1A 00 00 00 00 00 00 00 00 00
1A 00 BE 47 77 95 0D B6 2F B4 33 4D 64 95 B4 11 3D 99 7F 36 C8 8D BD 8A A9 DD D8 7C BD 38 3F B2 FE E3 11 79 A8 69 D5 D9 88 59 A5
8F EB CB B5 1D D1 29 69 E0 1A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03 03 1D 03 78 00 78 00 AC
00 00 04 04 1D 03 78 00 78 00 58 03 40 01 72 03 1B 00 49 15 FB A8 0F 86 09 DB 41 F5 E4 4B 1A 2D 32 21 69 6B A5 EB E6 98 6F 75 55
9B 1A 12 97 54 C7 D4 32 8C 8F 62 B1 EC 99 FF CD 87 91 EF BB 40 1A E6 F1 00 1A EC 30 A0 E7 58 D4 69 56 EA E3 F8 98 7B 84 13 4E 7A
AF E7 D7 79 60 B6 65 EF AC FA 90 9F F6 A0 37 AC 5D EE 86 A4 B5 02 4D 7B AC 24 3F D2 71 A9 EB 95 80 FF 00 00 00 00 00 00 00 00 00
03 00 84 D2 5D 6D 70 C2 83 E7 4B 8A 5B B2 D5 05 27 61 90 73 D8 92 BE 5D 95 78 E1 03 EE 62 63 8A E6 32 BF 6D 84 68 CD E5 C0 92 00
00 00 DA 21 00 00 00 00 08 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0A 0A 00 00 00 00 00 00 6E
F9 D3 13 B0 7C 37 B5 04 44 FB 5A 9B 07 80 97 A8 C6 DA 8C 58 9E 05 3F 34 6C 9F 6B 4D F6 EA 73 4B D2 A6 24 65 50 B1 AE D7 53 1F DE
00 00 00 00 00 00 00 00 00 00 00 01 AB 03 16 00 18 00 00 00 E8 00 CB 03 1A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
A6 AB 63 8D 30 C2 85 EF 4D E6 42 06 A0 74 C9 CB DC 2C C6 C3 4D 17 80 8D BF 20 B3 DE E0 51 00 FC D8 C7 A6 9A 20 9D CA 0B 12 FF E4
A8 CB 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03 03 1D 03 78 00 78 00 AC 00 69 01 49 03 1B 00 00
78 00 58 03 40 01 72 03 1B 00 99 3E C2 91 48 C7 23 45 F9 52 C5 3C 58 7D AC FC F6 64 F6 3B 24 A0 D9 6A E4 1A B0 E7 4E 97 30 1C 4C
A8 EA E8 B1 67 B7 BA 8F 17 1D 68 1D 8C CB 4E 86 30 3A F0 85 17 1A 71 32 2B BD A0 74 97 8B AE B0 9A 5A E5 14 77 4E 19 56 01 38 58
65 DC 33 18 49 43 E7 A4 0A FC 49 78 00 35 DC 29 1D B8 03 09 A2 D1 66 85 D0 44 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
86 6E 2F 14 E7 EB E9 C3 97 03 EE 28 98 5F 06 95 40 51 1B 28 25 0A E6 44 43 4D 80 1B 0D A2 6F 4F C0 82 00 00 00 00 00 00 00 00 00
08 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0A 0A 00 00 00 00 00 6E 22 00 00 00 00 08 00 28
7D F0 28 70 21 00 7C FD 7D F0 28 70 21 40 3C FD 75 F0 28 70 61 00 3C FC 7D F8 28 40 21 40 7C FC 7D F0 28 40 61 00 7C FC 7D F0 28
7D F0 28 40 21 40 7C FD 7D F0 28 70 21 00 7C FD 7D F8 28 70 21 00 7C FD 7D F0 28 60 21 00 7C FC 75 F0 28 70 21 00 7C 7D 7D F0 28
7D F4 28 70 21 00 3C 7D 7D F0 28 70 61 00 7C 7D 75 F0 28 70 21 00 3C FC 7D F0 28 70 61 40 7C FD 7D F0 28 60 21 40 3C FC 7D F0 28
7D F8 28 70 21 00 7C 7D 75 F8 28 60 61 00 3E FC 7D F0 28 70 21 00 7C FC 75 F0 28 70 21 00 3E FD 75 F0 28 70 61 00 7C FD 7D F8 28
7D F0 28 70 21 00 7C FC 7D F8 28 70 21 40 7C FD 7D F0 28 60 21 00 7C FD 7D F0 28 60 61 00 3C FD FD F0 28 60 21 00 7C FC 7D F8 28
7D F0 28 50 61 40 7C FC FD F8 28 60 21 00 3C 7D 7D F0 28 70 21 40 7C FD 7D F8 28 60 21 00 7C FD 7D F0 28 40 21 00 7C FD 7D F0 28
7D F0 28 60 21 00 3C FD 7D F0 2C 40 21 00 7D FD 7D F0 28 70 21 00 7C FC 7D F0 28 70 61 00 7C FC 7D F0 28 60 21 00 7C 7C 7D F0 28
7D F0 28 40 25 00 7C FD 75 F0 28 60 21 00 7C FD 7D F0 28 40 21 00 7C FD 75 F0 28 40 21 00 3E FC 7D F0 28 50 21 00 7C 7D F5 F0 28
7D F0 28 60 21 00 3E FC 7D F8 28 60 21 00 7C FC 7D F4 28 70 21 00 7C FD 7D F8 28 70 21 10 7C FD 7D F0 28 60 21 00 3C FC 7D F0 2C
7D F0 28 40 21 00 7C FD 7D F0 28 70 21 00 3C FC 75 F0 28 40 21 10 7C FD 75 F0 28 60 21 00 7C FD 7D F0 2C 70 21 00 7C FD 7D F0 28
75 F4 28 68 21 00 3C FC 7D F0 2C 50 21 00 3C FD 7D F0 0E 00 00 7D
```



```
E9 8F 2E C1 5D F7 11 5B 28 1C F7 CD B1 DB E5 96 0F 35 AE A9 D5 B0 50 DD FD 96 F3 6B 00 7C 6F C0 FF 92 ED 2A 7C 76 FF 43 87 47 83
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01 AB 03 16 00 18 00 00 00 E8 00 CB 03 1A 00 00 00 00 00 00 00 00 00 00
1A 00 BE 47 77 95 0D B6 2F B4 33 4D 64 95 B4 11 3D 99 7F 36 C8 8D BD 8A A9 DD D8 7C BD 38 3F B2 FE E3 11 79 A8 69 D5 D9 88 59 A5
8F EB CB B5 1D D1 29 69 E0 1A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03 03 1D 03 78 00 78 00 AC
00 00 04 04 1D 03 78 00 78 00 58 03 40 01 72 03 1B 00 49 15 FB A8 0F 86 09 DB 41 F5 E4 4B 1A 2D 32 21 69 6B A5 EB E6 98 6F 75 55
9B 1A 12 97 54 C7 D4 32 8C 8F 62 B1 EC 99 FF CD 87 91 EF BB 40 1A E6 F1 00 1A EC 30 A0 E7 58 D4 69 56 EA E3 F8 98 7B 84 13 4E 7A
AF E7 D7 79 60 B6 65 EF AC FA 90 9F F6 A0 37 AC 5D EE 86 A4 B5 02 4D 7B AC 24 3F D2 71 A9 EB 95 80 FF 00 00 00 00 00 00 00 00 00
03 00 84 D2 5D 6D 70 C2 83 E7 4B 8A 5B B2 D5 05 27 61 90 73 D8 92 BE 5D 95 78 E1 03 EE 62 63 8A E6 32 BF 6D 84 68 CD E5 C0 92 00
00 00 DA 21 00 00 00 00 08 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0A 0A 00 00 00 00 00 00 6E
F9 D3 13 B0 7C 37 B5 04 44 FB 5A 9B 07 80 97 A8 C6 DA 8C 58 9E 05 3F 34 6C 9F 6B 4D F6 EA 73 4B D2 A6 24 65 50 B1 AE D7 53 1F DE
00 00 00 00 00 00 00 00 00 00 00 01 AB 03 16 00 18 00 00 00 E8 00 CB 03 1A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
A6 AB 63 8D 30 C2 85 EF 4D E6 42 06 A0 74 C9 CB DC 2C C6 C3 4D 17 80 8D BF 20 B3 DE E0 51 00 FC D8 C7 A6 9A 20 9D CA 0B 12 FF E4
A8 CB 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03 03 1D 03 78 00 78 00 AC 00 69 01 49 03 1B 00 00
78 00 58 03 40 01 72 03 1B 00 99 3E C2 91 48 C7 23 45 F9 52 C5 3C 58 7D AC FC F6 64 F6 3B 24 A0 D9 6A E4 1A B0 E7 4E 97 30 1C 4C
A8 EA E8 B1 67 B7 BA 8F 17 1D 68 1D 8C CB 4E 86 30 3A F0 85 17 1A 71 32 2B BD A0 74 97 8B AE B0 9A 5A E5 14 77 4E 19 56 01 38 58
65 DC 33 18 49 43 E7 A4 0A FC 49 78 00 35 DC 29 1D B8 03 09 A2 D1 66 85 D0 44 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
86 6E 2F 14 E7 EB E9 C3 97 03 EE 28 98 5F 06 95 40 51 1B 28 25 0A E6 44 43 4D 80 1B 0D A2 6F 4F C0 82 00 00 00 00 00 00 00 00 00
08 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0A 0A 00 00 00 00 00 6E 22 00 00 00 00 08 00 28
7D F0 28 70 21 00 7C FD 7D F0 28 70 21 40 3C FD 75 F0 28 70 61 00 3C FC 7D F8 28 40 21 40 7C FC 7D F0 28 40 61 00 7C FC 7D F0 28
7D F0 28 40 21 40 7C FD 7D F0 28 70 21 00 7C FD 7D F8 28 70 21 00 7C FD 7D F0 28 60 21 00 7C FC 75 F0 28 70 21 00 7C 7D 7D F0 28
7D F4 28 70 21 00 3C 7D 7D F0 28 70 61 00 7C 7D 75 F0 28 70 21 00 3C FC 7D F0 28 70 61 40 7C FD 7D F0 28 60 21 40 3C FC 7D F0 28
7D F8 28 70 21 00 7C 7D 75 F8 28 60 61 00 3E FC 7D F0 28 70 21 00 7C FC 75 F0 28 70 21 00 3E FD 75 F0 28 70 61 00 7C FD 7D F8 28
7D F0 28 70 21 00 7C FC 7D F8 28 70 21 40 7C FD 7D F0 28 60 21 00 7C FD 7D F0 28 60 61 00 3C FD FD F0 28 60 21 00 7C FC 7D F8 28
7D F0 28 50 61 40 7C FC FD F8 28 60 21 00 3C 7D 7D F0 28 70 21 40 7C FD 7D F8 28 60 21 00 7C FD 7D F0 28 40 21 00 7C FD 7D F0 28
7D F0 28 60 21 00 3C FD 7D F0 2C 40 21 00 7D FD 7D F0 28 70 21 00 7C FC 7D F0 28 70 61 00 7C FC 7D F0 28 60 21 00 7C 7C 7D F0 28
7D F0 28 40 25 00 7C FD 75 F0 28 60 21 00 7C FD 7D F0 28 40 21 00 7C FD 75 F0 28 40 21 00 3E FC 7D F0 28 50 21 00 7C 7D F5 F0 28
7D F0 28 60 21 00 3E FC 7D F8 28 60 21 00 7C FC 7D F4 28 70 21 00 7C FD 7D F8 28 70 21 10 7C FD 7D F0 28 60 21 00 3C FC 7D F0 2C
7D F0 28 40 21 00 7C FD 7D F0 28 70 21 00 3C FC 75 F0 28 40 21 10 7C FD 75 F0 28 60 21 00 7C FD 7D F0 2C 70 21 00 7C FD 7D F0 28
75 F4 28 68 21 00 3C FC 7D F0 2C 50 21 00 3C FD 7D F0 0E 00 00 7D
```

That's obviously the cmd packet!

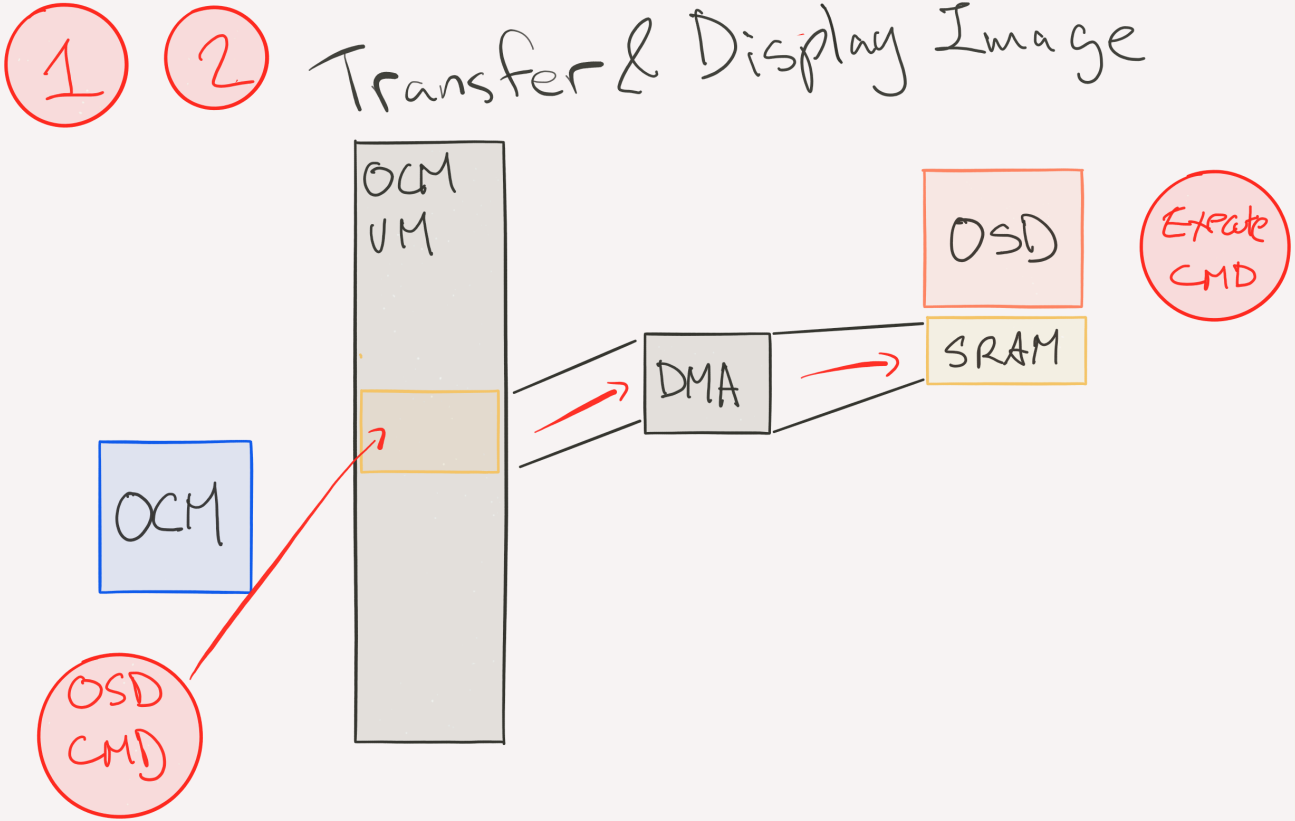


OSD COMMAND CONTROL STRUCTURE

40 bytes
in total

Patterns and Hue ???	38
Transparency & bit per pixel mode	37
Y Coordinate	36
Y Size	34
Pointer in SDRAM	32
Data Expansion	30
X Size	28
X Coordinate	26
Color	24
\x00...\x00	0

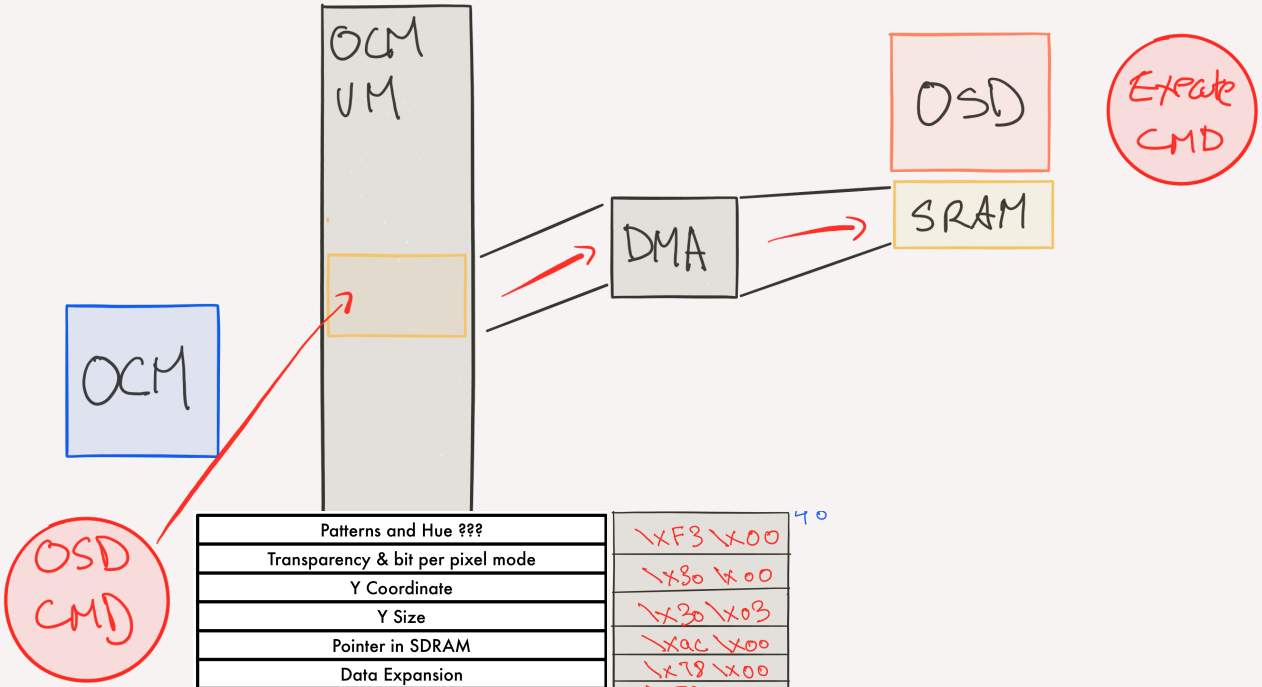




Answered Question 1 & 2



1 2 Transfer & Display Image



Answered Question 1 & 2



```
segF000:7A12 sdram_read_variable_offset proc far      ; CODE XREF: segF000:00E1|j
segF000:7A12                                          ; sdram_read_0_rd_offset+1B|p
segF000:7A12
segF000:7A12 arg_0           = word ptr  6
segF000:7A12 arg_2           = word ptr  8
segF000:7A12 arg_4           = word ptr  0Ah
segF000:7A12 arg_6           = word ptr  0Ch
segF000:7A12 arg_8           = word ptr  0Eh
segF000:7A12 arg_A           = word ptr  10h
segF000:7A12 arg_C           = word ptr  12h
segF000:7A12 arg_E           = word ptr  14h
segF000:7A12
segF000:7A12                 push    bp
segF000:7A13                 mov     bp, sp
segF000:7A15                 push    si
segF000:7A16                 push    di
segF000:7A17                 push    ds
segF000:7A18                 mov     ax, 0
segF000:7A1B                 mov     ds, ax
segF000:7A1D                 mov     si, [bp+arg_C]  ; 415e
segF000:7A20                 cmp     [bp+arg_4], 0
segF000:7A24                 jz      short loc_7A39
segF000:7A26                 cmp     [bp+arg_6], 0
segF000:7A2A                 jz      short loc_7A39
segF000:7A2C                 cmp     [bp+arg_8], 0
segF000:7A30                 jz      short loc_7A39
segF000:7A32                 test    [bp+arg_6], 1
segF000:7A37                 jz      short loc_7A3F
segF000:7A39
segF000:7A39 loc_7A39:                                ; CODE XREF: sdram_read_variable_offset+
segF000:7A39                                          ; sdram_read_variable_offset+18|j ...
segF000:7A39                 mov     ax, 4
segF000:7A3C                 jmp     loc_7B2D
segF000:7A3F ; ---------------------------------------------------------------------------
segF000:7A3F
segF000:7A3F loc_7A3F:                                ; CODE XREF: sdram_read_variable_offset+
segF000:7A3F                 mov     ax, [bp+arg_6]
segF000:7A42                 shr     ax, 1
segF000:7A44                 imul    [bp+arg_8]
segF000:7A47                 mov     di, ax          ; 0x20
segF000:7A49                 push    cs
segF000:7A4A                 call    near ptr sub_7627
segF000:7A4D                 push    [bp+arg_A]
segF000:7A50                 push    [bp+arg_8]
segF000:7A53                 push    [bp+arg_6]
segF000:7A56                 push    [bp+arg_4]
segF000:7A59                 push    [bp+arg_2]
segF000:7A5C                 push    [bp+arg_0]
segF000:7A5F                 call    dword ptr ds:loc_22E+2 ; 76c1
segF000:7A63                 add     sp, 0Ch
segF000:7A66                 or      byte ptr ds:loc_D3E1+1, 1 ; start sdram read
segF000:7A6B                 mov     es, [bp+arg_E]  ; 415e
segF000:7A6E                 jmp     short loc_7AB3
```

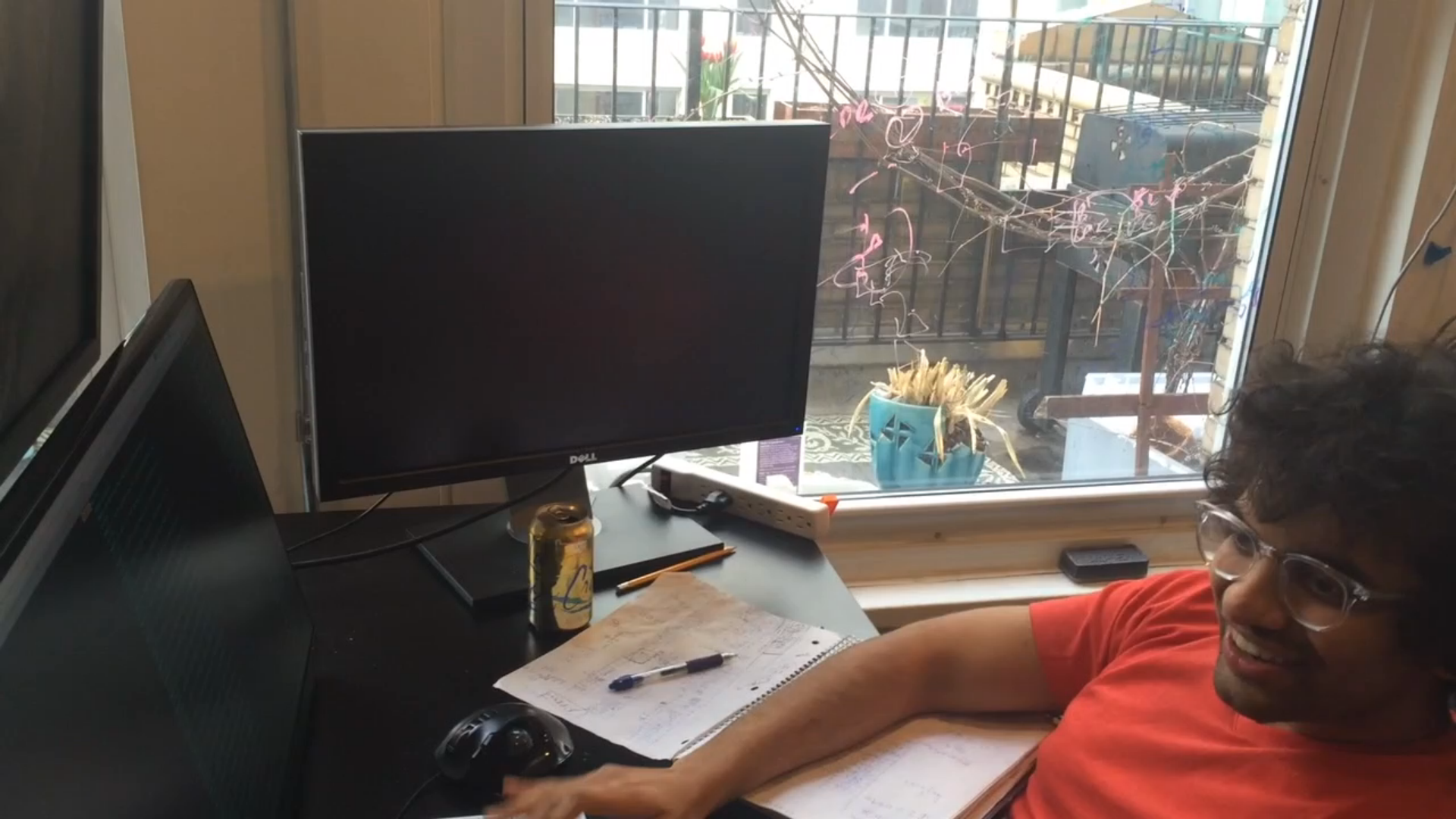
Transfers the image



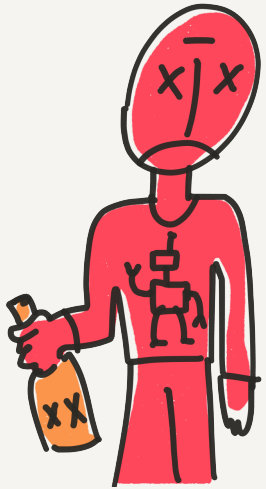
```
segF000:77F0
segF000:77F0 sdram_write_variable_offset proc far    ; CODE XREF: segF000:loc_DE1j
segF000:77F0
segF000:77F0 arg_0           = word ptr  6
segF000:77F0 arg_2           = word ptr  8
segF000:77F0 arg_4           = word ptr  0Ah
segF000:77F0 arg_6           = word ptr  0Ch
segF000:77F0 arg_8           = word ptr  0Eh
segF000:77F0 arg_A           = word ptr  10h
segF000:77F0 arg_C           = word ptr  12h
segF000:77F0 arg_E           = word ptr  14h
segF000:77F0 arg_10          = word ptr  16h
segF000:77F0 arg_12          = word ptr  18h
segF000:77F0 arg_14          = byte ptr  1Ah
segF000:77F0 arg_16          = word ptr  1Ch
segF000:77F0
segF000:77F0                 push    bp
segF000:77F1                 mov     bp, sp
segF000:77F3                 push    si
segF000:77F4                 push    di
segF000:77F5                 push    ds
segF000:77F6                 mov     ax, 0
segF000:77F9                 mov     ds, ax
segF000:77FB                 mov     si, [bp+arg_C]  ; buffer
segF000:77FE                 cmp     [bp+arg_4], 0
segF000:7802                 jz      short loc_783A
segF000:7804                 cmp     [bp+arg_6], 0
segF000:7808                 jz      short loc_783A
segF000:780A                 cmp     [bp+arg_8], 0
segF000:780E                 jz      short loc_783A
segF000:7810                 cmp     [bp+arg_16], 0
segF000:7814                 jz      short loc_783A
segF000:7816                 mov     ax, [bp+arg_10]
segF000:7819                 or      ax, [bp+arg_12]
segF000:781C                 jz      short loc_7840
segF000:781E                 mov     al, [bp+arg_14]
segF000:7821                 cbw
segF000:7822                 and     ax, 78h
segF000:7825                 cmp     ax, 60h ; '`'
segF000:7828                 jl      short loc_7840
segF000:782A                 push    [bp+arg_12]
segF000:782D                 push    [bp+arg_10]
segF000:7830                 call    sub_775B        ; checking bounds with 0x7FFF
segF000:7833                 add     sp, 4
segF000:7836                 or      ax, ax
segF000:7838                 jnz     short loc_7840
segF000:783A
segF000:783A loc_783A:                               ; CODE XREF: sdram_write_variable_offset+12Ij
segF000:783A                                         ; sdram_write_variable_offset+18Ij ...
segF000:783A                 mov     ax, 4
segF000:783D                 jmp     loc_79DD
segF000:7840 ; ---------------------------------------------------------------------------
segF000:7840
segF000:7840 loc_7840:                               ; CODE XREF: sdram_write_variable_offset+2CIj
segF000:7840                                         ; sdram_write_variable_offset+38Ij ...
segF000:7840                 push    cs
segF000:7841                 call    near ptr sub_75E2
segF000:7844                 push    [bp+arg_A]
segF000:7847                 push    [bp+arg_8]
segF000:784A                 push    [bp+arg_6]
segF000:784D                 push    [bp+arg_4]
segF000:7850                 push    [bp+arg_2]
segF000:7853                 push    [bp+arg_0]
segF000:7856                 call    dword ptr ds:loc_22B+1 ; 7686
segF000:785A                 add     sp, 0Ch
segF000:785D                 mov     ax, [bp+arg_10]
segF000:7860                 or      ax, [bp+arg_12]
segF000:7863                 jnz     short loc_78DC
segF000:7865                 mov     ax, [bp+arg_16]
segF000:7868                 mul     [bp+arg_8]
```

Transfers the image





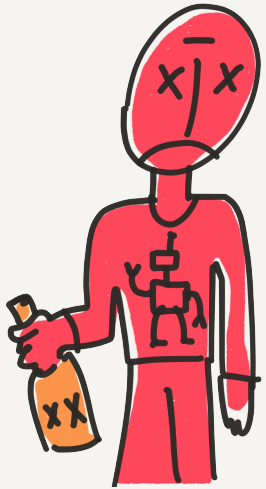
3 ... what is a color





Box with

Color 0
Color 1
Color 2
Color 3

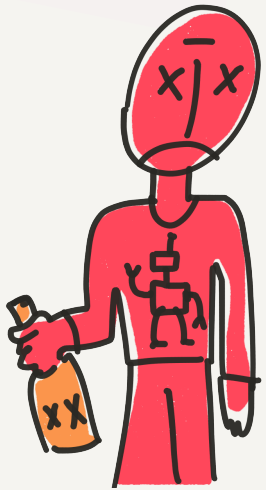


Science Time!

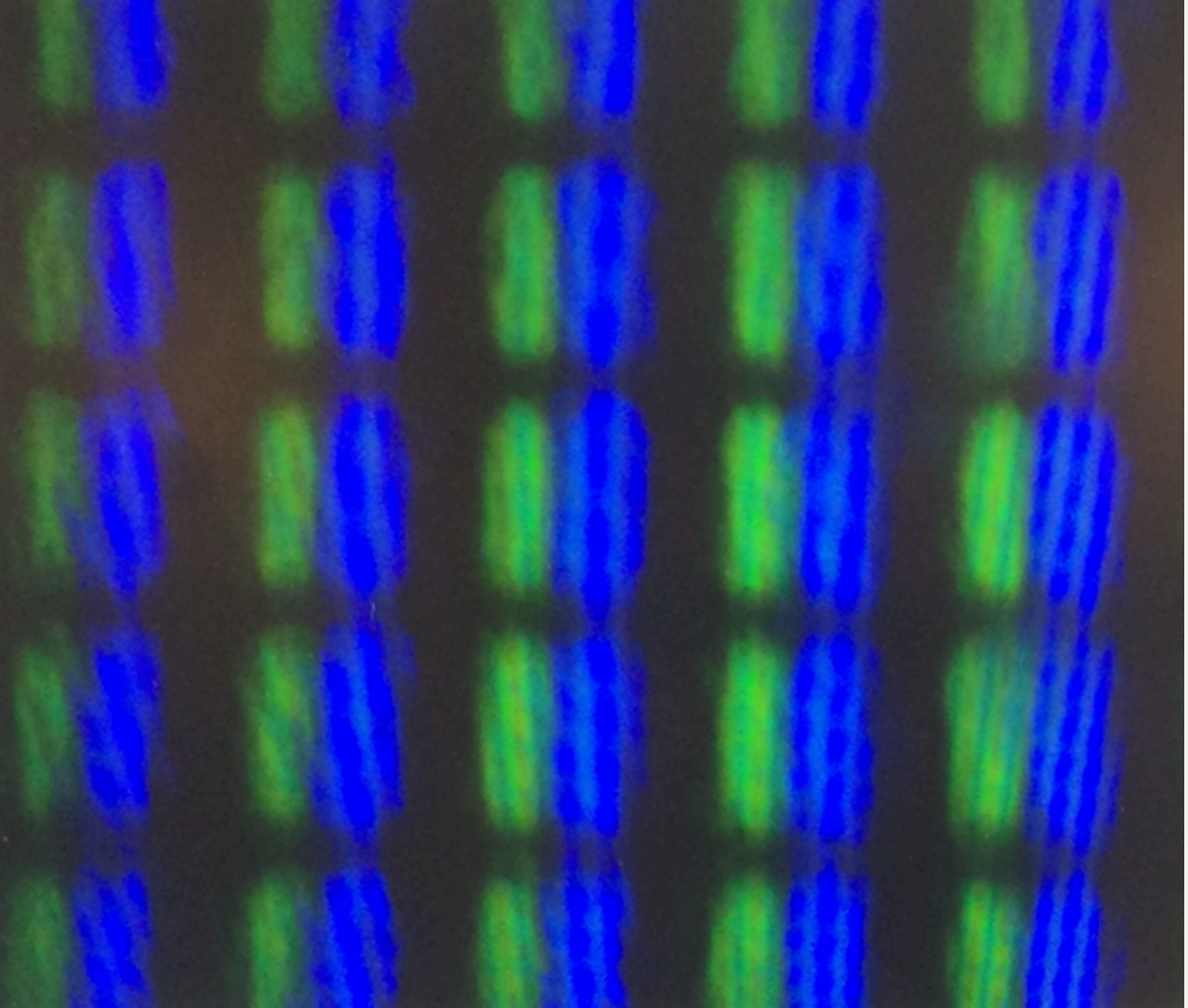




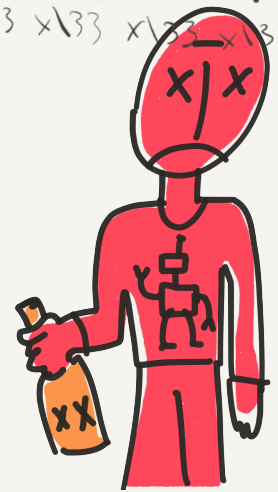
Science Time!

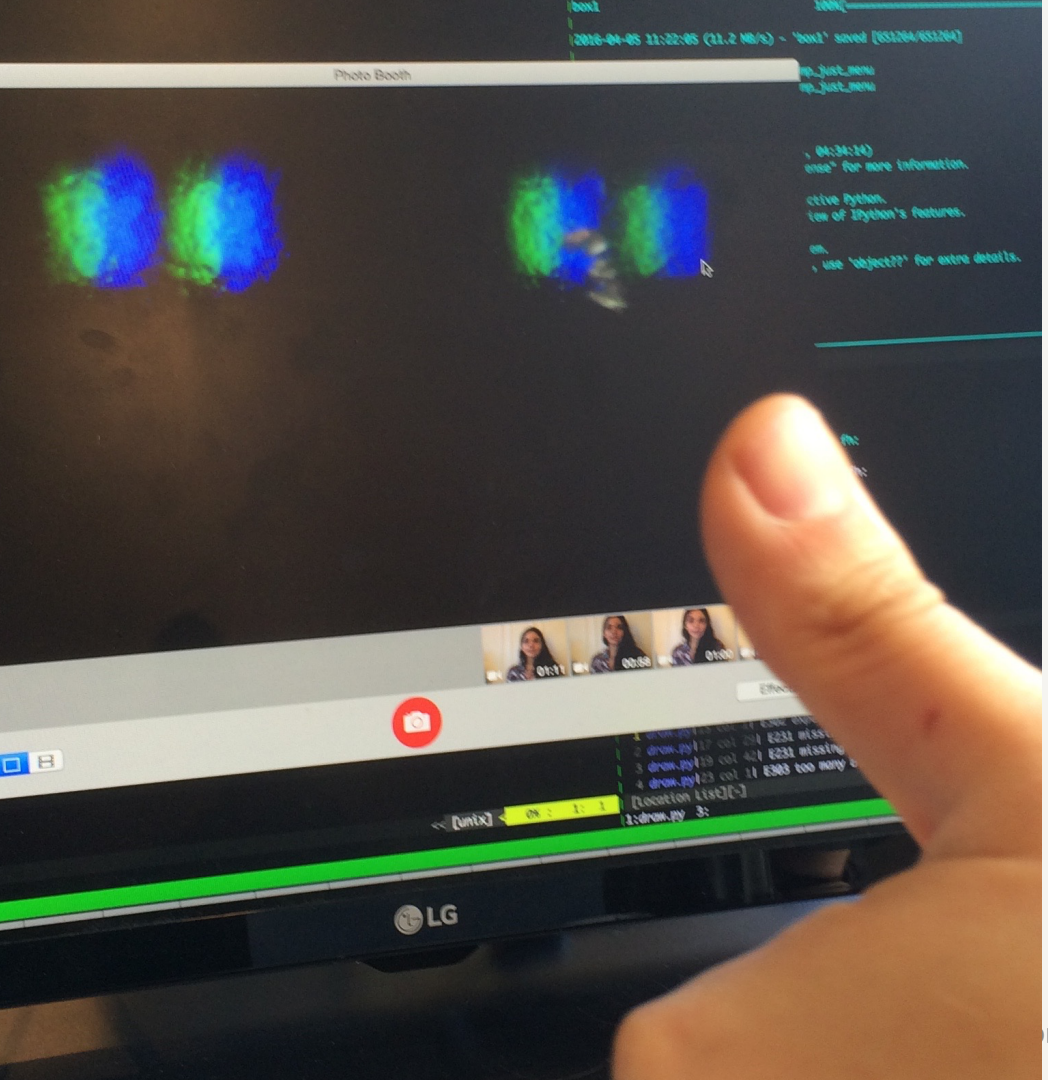


Monitor Darkly

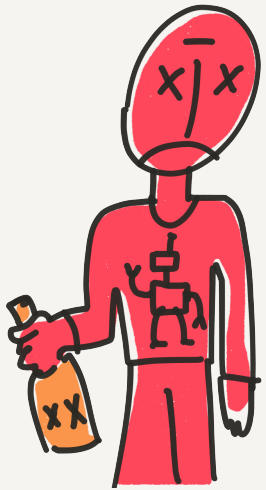


x133 x133 x133 x133 x133 x133 x133
x133 x133 x133 x133 x133 x133 x133
x133 x133 x133 x133 x133 x133 x133
x133 x133 x133 x133 x133 x133 x133
x133 x133 x133 x133 x133 x133 x133
x133 x133 x133 x133 x133 x133 x133
x133 x133 x133 x133 x133 x133 x133
x133 x133 x133 x133 x133 x133 x133

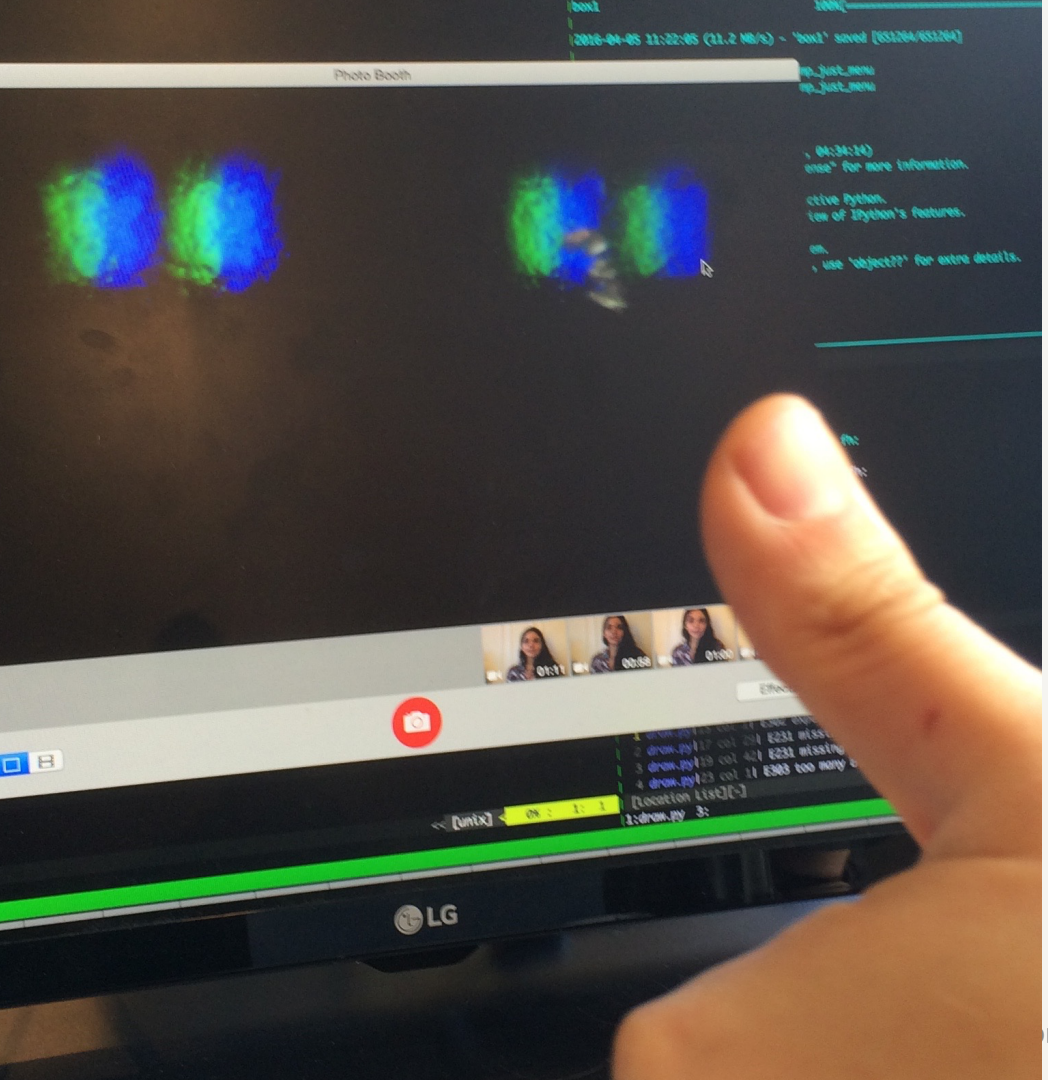




x\33 x\00 x\33 x\00

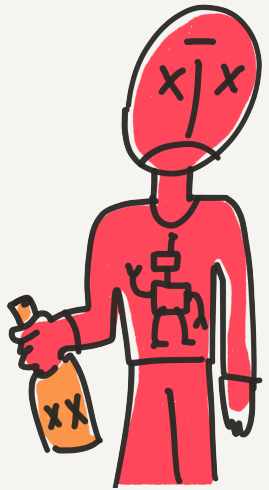


r Darkly



x133 x100 x133 x100

How Many Bits Per Pixel?



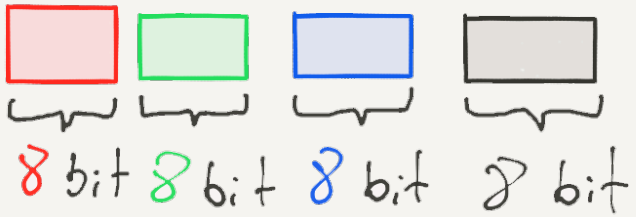
4 bits per pixel?!



4 bits per pixel?!

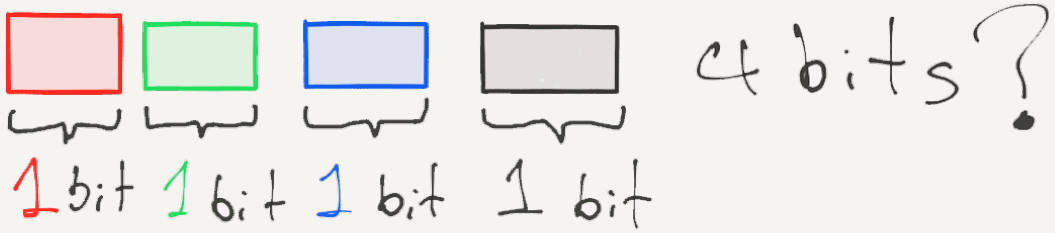
How do you encode a COLOR with 4 bits?





32 bits




1 bit
1 bit
1 bit
1 bit
4 bits?


XX

Color
Look
Up
Table!

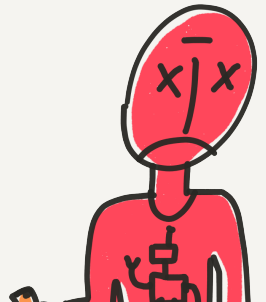


Color
Look
Up
Table!

Index	Color (32-bit)
0	
1	
2	
3	
4	
5	
6	
...	



where is the CLUT?




Dump

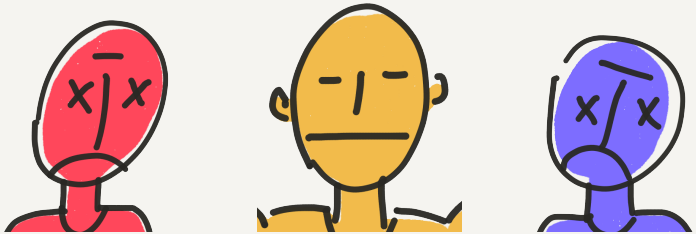

Drink
Dump
Drink

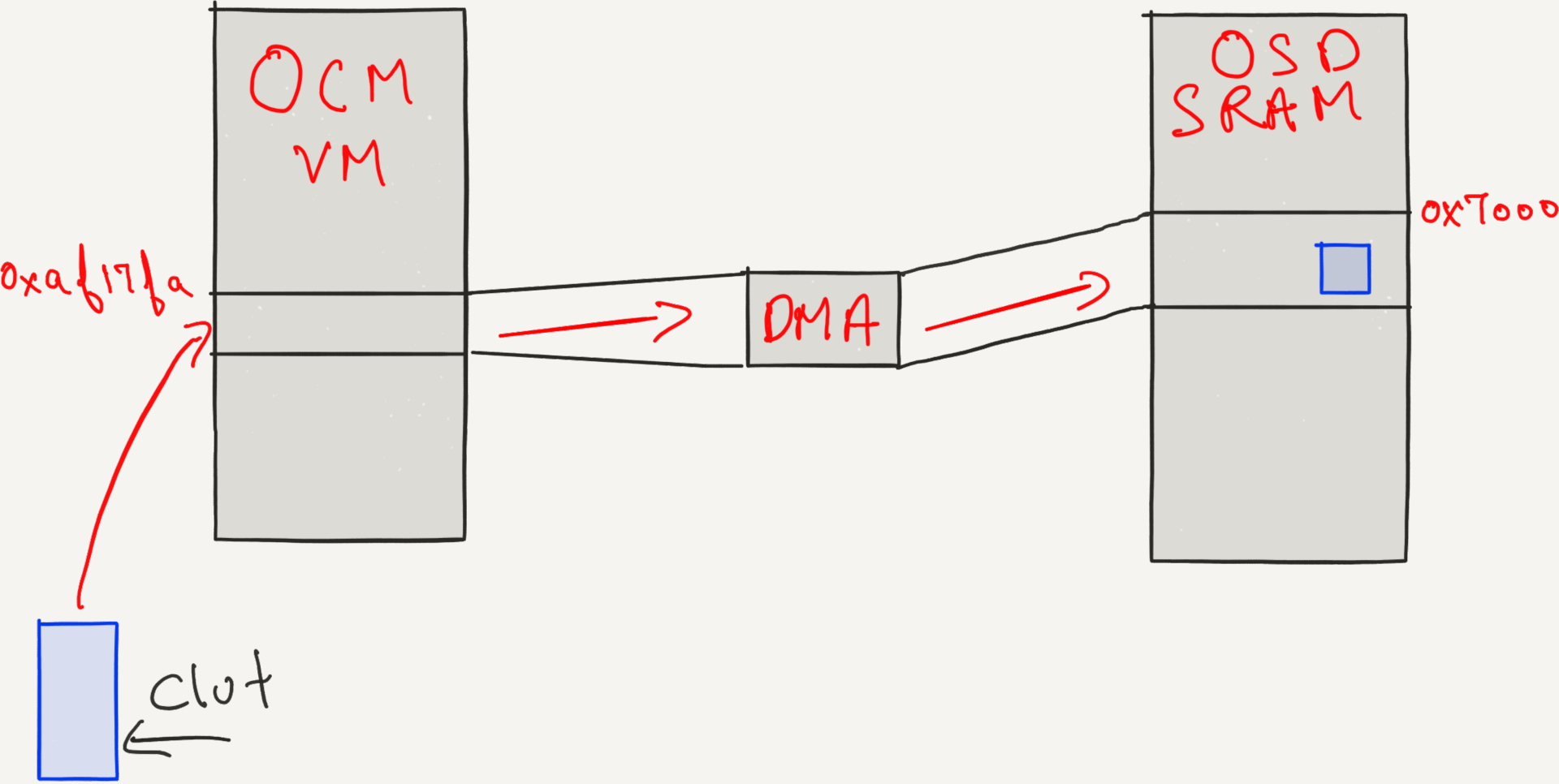

DumpDump
Drink
Drink
Drink
Drink


Dump Dump
Dump
Dump
Dump
Drink Drink
Drink
Drink
Drink
Drink
Drink
Drink

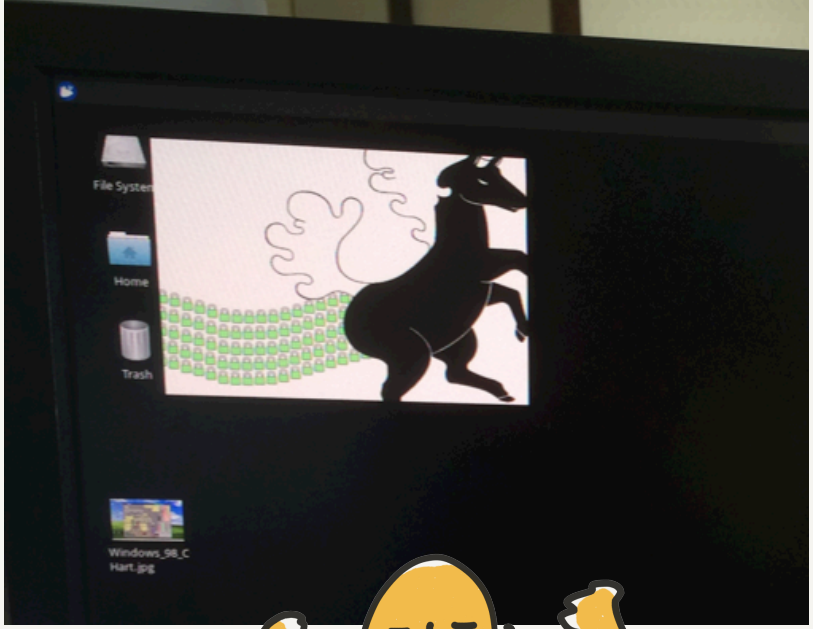
like 2 days later....

oh look at that!




OCM
VM
OSD
SRAM
0xaf17fa
DMA
0x7000
clut

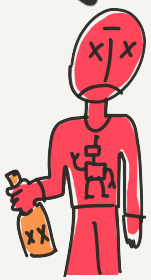



File System
Home
Trash
Windows_98_C
Hart.jpg





not working



We need more colors!

#!*#? huh?



4.16 OSD

The gm1601 has a fully programmable, true color bitmapped OSD controller capable of displaying up to 12 “tiles” or bitmap windows on the display. The individual tiles are programmable for location, size, and bits per pixel, and have a precedence determining which tiles appear when overlapping occurs on the display. Tile data is stored in the external framestore memory by the system controller in either: 1, 2, 4, or 8 bit per pixel format. On-chip table registers point to the start of tiles in external memory. Two programmable on-chip 256 x 24-bit color lookup tables are provided to map the OSD pixels onto a true 24 bit color space using the first table for one mapping, and the second table for an alternate mapping.



VMware Fusion File Edit View Virtual Machine Window Help

SecOps

Google Chrome

STDP8028-AB_Regis ×

www.doc88.com/p-2973089571270.html

下载 收藏 392 / 705

3.25 OCM BreakPoints and Code-Patch

These are 4 sets of registers for use in debug and patching of OCM mask-rom code. The OCM address for any memory access (code read, or data r/w) is compared, with a selectable mask of the low bits, to a "breakpoint" address. When the OCM address matches, either an BP_IRQ may be generated or the OCM addr ess will be substituted with a new address to "redirect" the OCM to a different part of the RAM or ROM (and patch over code bugs, or change ROM data tables). The BP_IRQ is available as an interrupt source to OCM_IRQ0 of the on chip microprocessor.

Breakpont vectoring is only possible for remapping an IROM to an ISRAM address.

0xD7C0	BP0_CTRL	RW
POD: 0x0000		
Firmware Breakpoint 0 Control		

BIT	BIT NAME	FUNCTION
2:0	BP0_MASK	Selects the number of LSBs of OCM address to mask for the Breakpoint compare, and also the LSBs of the OCM address to keep when using the address substitution. Defines a range of addresses of 2n for "redirecting" the OCM. (n = 0 to 5)
4:3	BP0_EN	Defines the operation of this Break-Point register set when the OCM address matches the Break-Point address: 00 = Disabled 01 = Substitute OCM address (over BP range) with the SUB_ADDR (and OCM LSBs) 10 = Set status and maskable IRQ on OCM writes to breakpoint address(es) 11 = Set status and maskable IRQ on OCM reads from breakpoint address(es)
15:5	Reserved	Reserved

0xD7C2	BP0_OCM_ADDR_hi	RW
POD: 0x0000		
Firmware Breakpoint 0 Address		

BIT	BIT NAME	FUNCTION
7:0	BP0_OCM_ADDR	23:16 bits of 20 bit breakpoint starting address range
15:8	Reserved	

BP FTW!



Interns! BP everything!





Dumped stack
heap

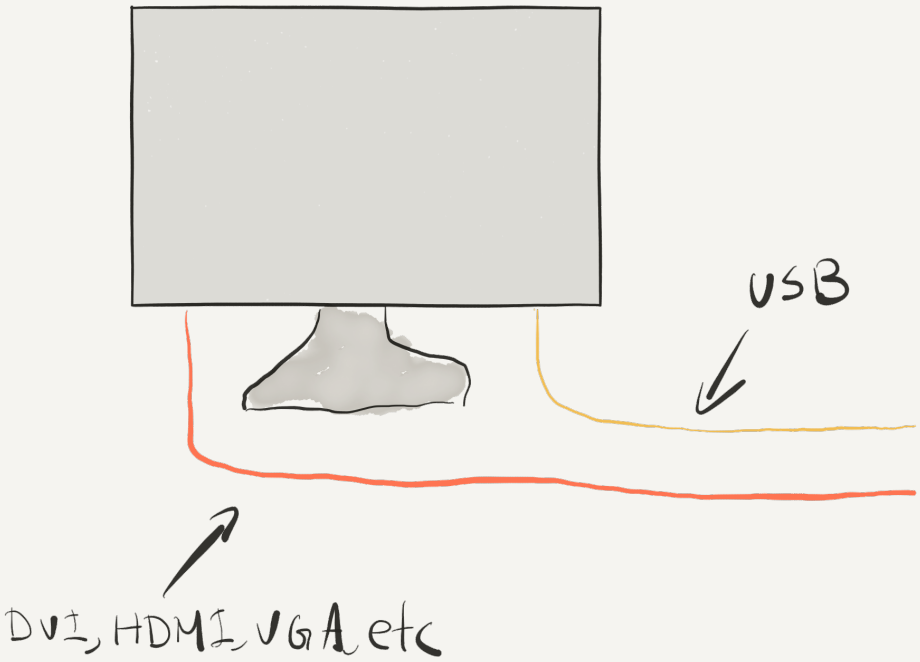
load clut, etc

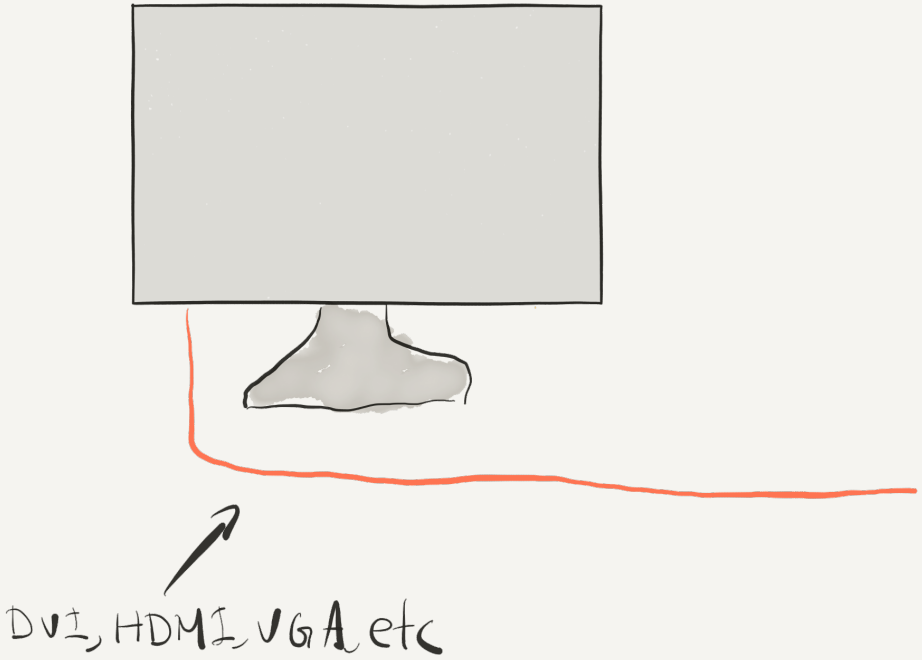
API exposed in PoC (on github)

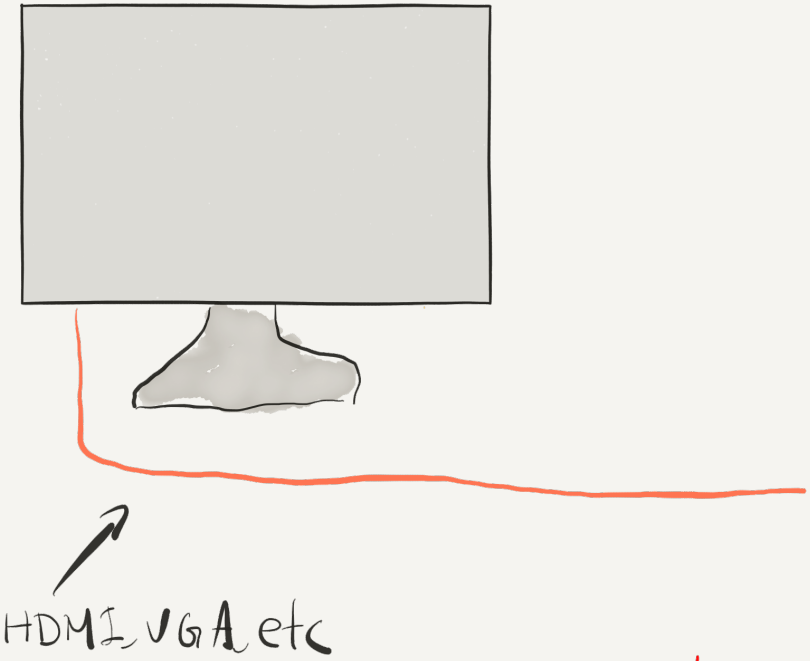


```
00A7FA14
00A7FA14
00A7FA14 ; Attributes: bp-based frame
00A7FA14
00A7FA14 grab_pixel proc far
00A7FA14
00A7FA14 arg_0= word ptr  6
00A7FA14 arg_2= word ptr  8
00A7FA14 arg_4= word ptr  0Ah
00A7FA14
00A7FA14 push    bp
00A7FA15 mov     bp, sp
00A7FA17 push    si
00A7FA18 mov     si, [bp+arg_4]   ;  memory address
00A7FA1B or      byte ptr ds:0D6D8h, 1
00A7FA20 mov     ax, [bp+arg_0]
00A7FA23 mov     ds:0D6DAh, ax
00A7FA26 mov     ax, [bp+arg_2]
00A7FA29 mov     ds:0D6DCh, ax
00A7FA2C push    2
00A7FA2E call    far ptr 0A51Ah:0F1h
00A7FA33 pop     cx
00A7FA34 mov     ax, ds:0D6DEh
00A7FA37 mov     [si], ax
00A7FA39 mov     ax, ds:0D6E0h
00A7FA3C mov     [si+2], ax
00A7FA3F mov     ax, ds:0D6E2h
00A7FA42 mov     [si+4], ax
00A7FA45 pop     si
00A7FA46 pop     bp
00A7FA47 retf
00A7FA47 grab_pixel endp
00A7FA47
```

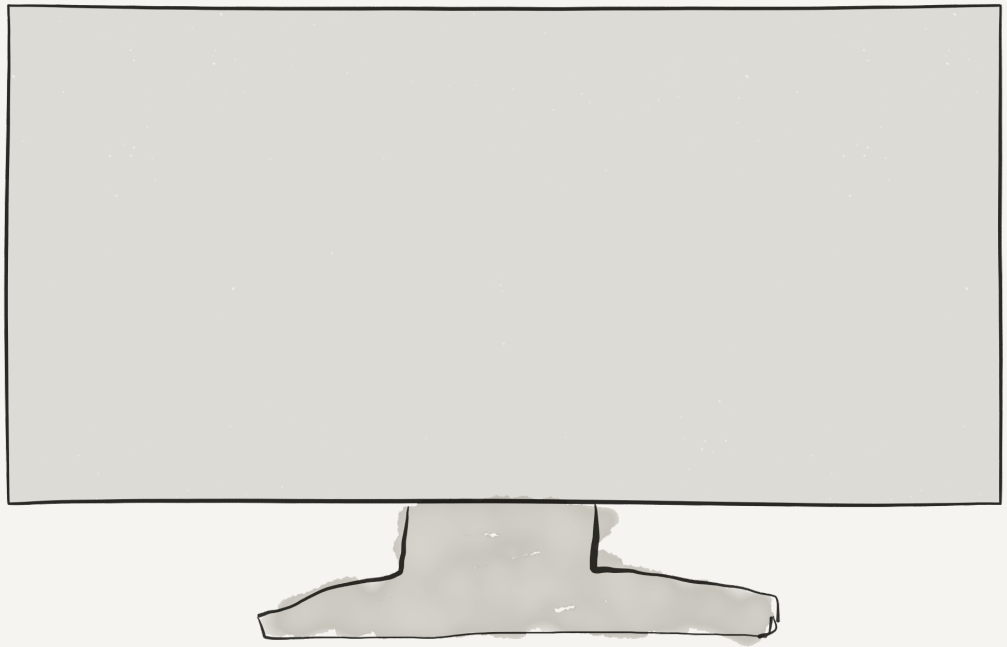



USB
DVI, HDMI, VGA etc

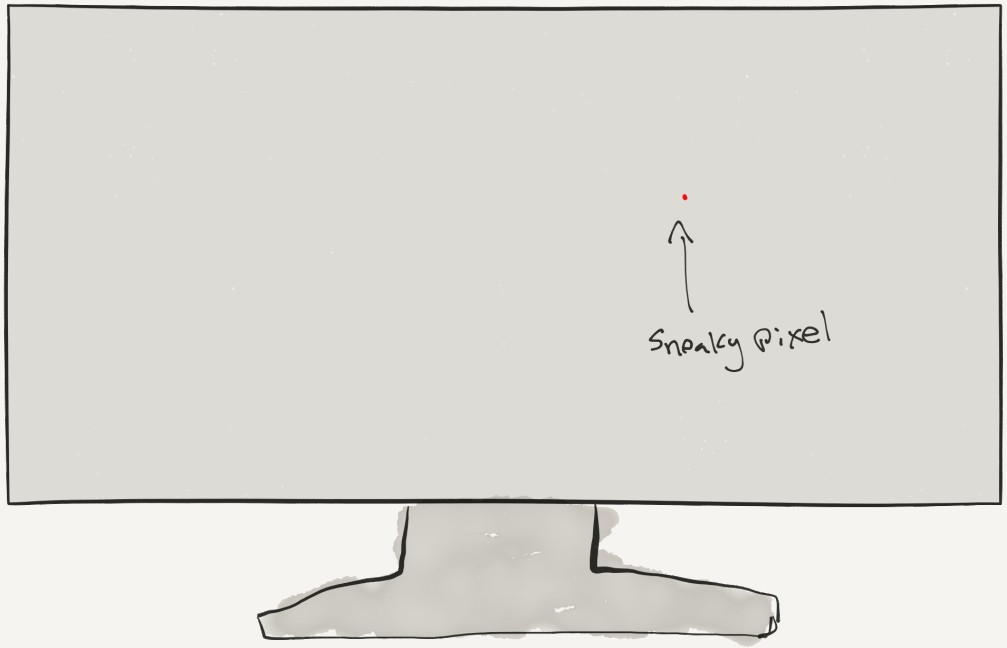

DVI, HDMI, VGA etc

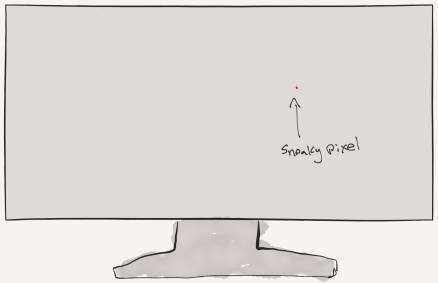

DVI, HDMI, VGA etc
code already on github !!

Monitor Implant!



Monitor Implant!

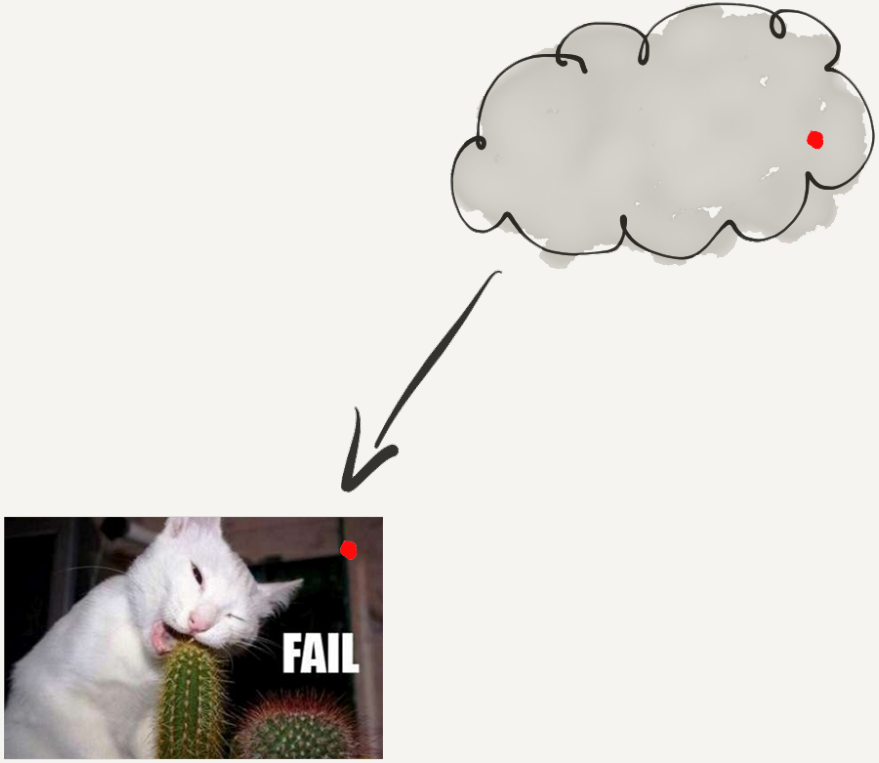


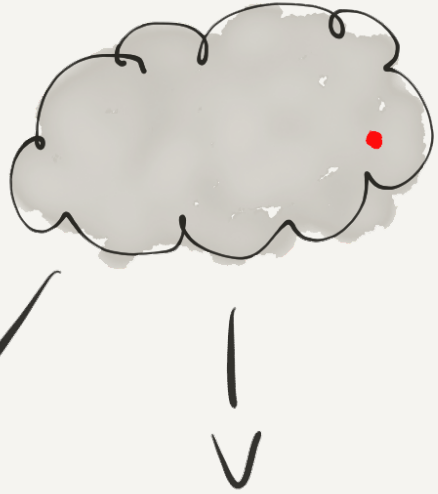


Cmd_type Data Data Data ....

8 bits 8 bits 8 bits




FAIL



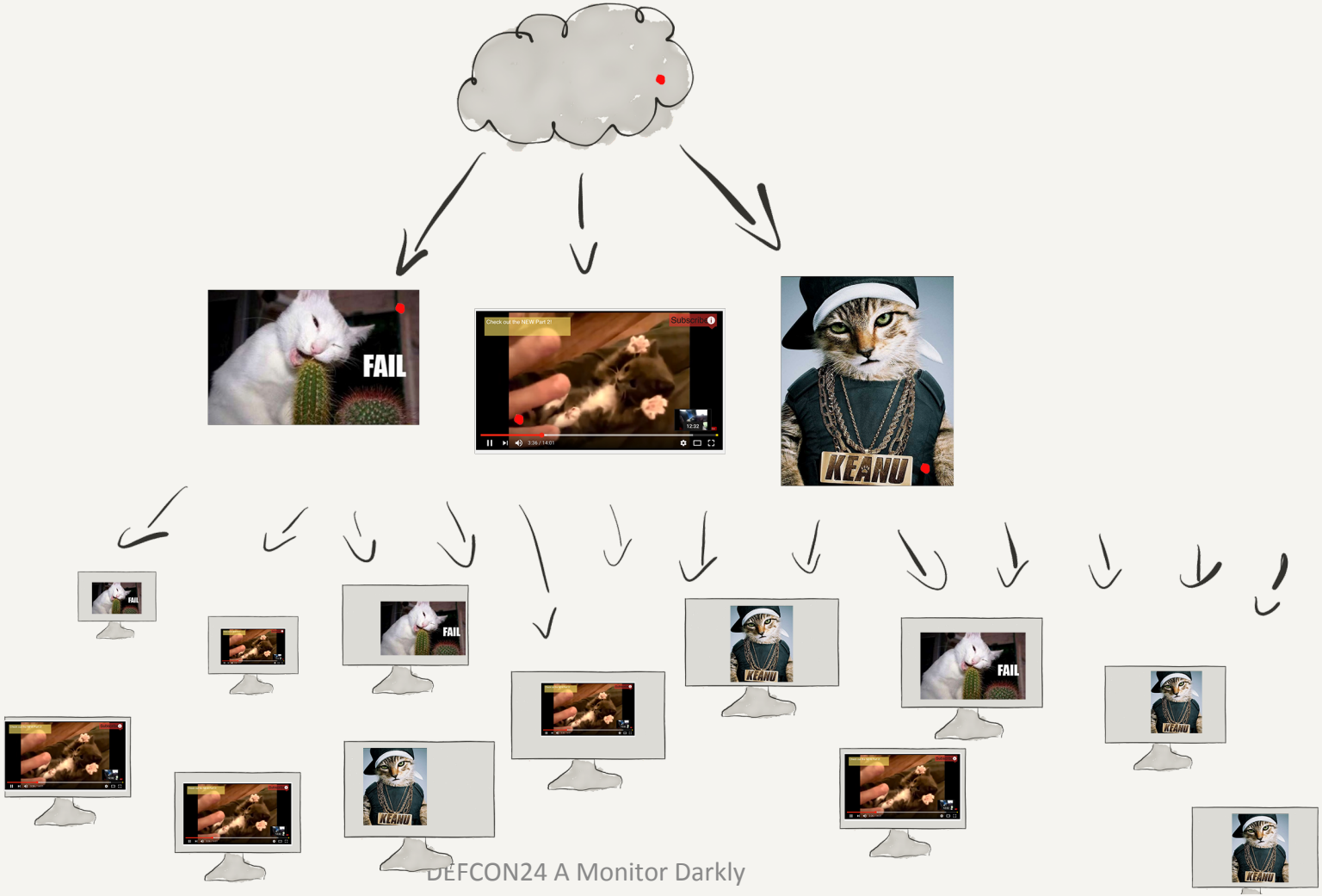

FAIL


Check out the NEW Part 2!
Subscribe
12:32
3:36 / 14:01

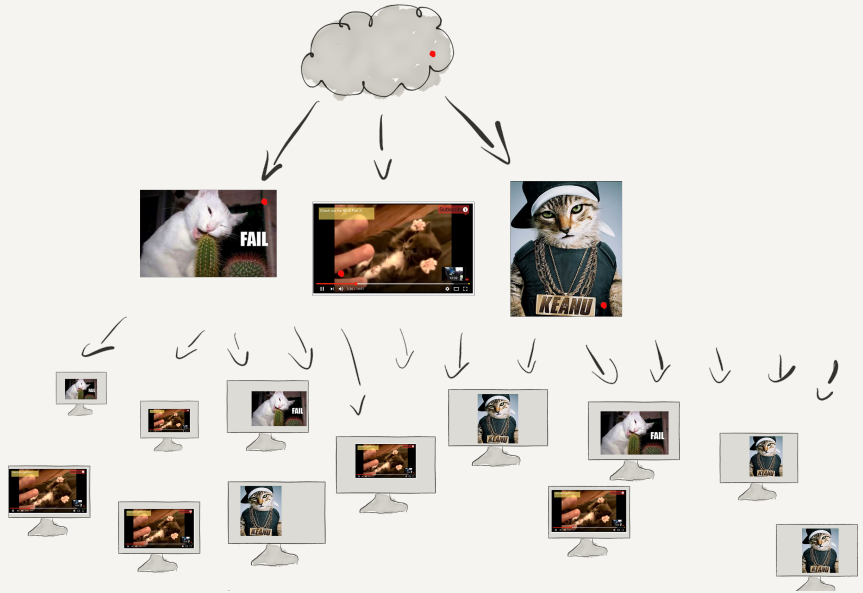

FAIL


Check out the NEW Part 2!
Subscribe
12:32
3:36 / 14:01


KEANU

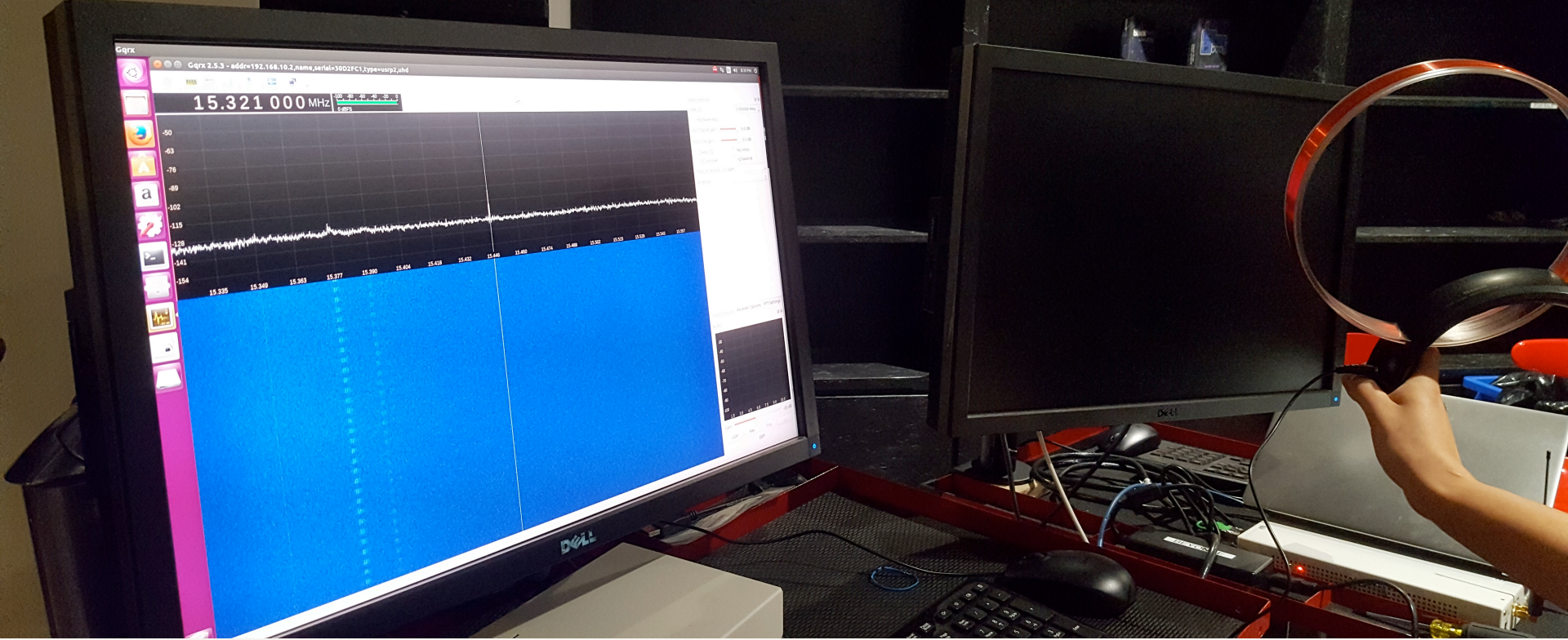

FAIL
Check out the NEW Part 2!
Subscribe
3:36 / 14:01
12:32
KEANU

Cat-Based World Domination Plan #1



In the end

1. change any pixel? ✓
2. see every pixel? ✓
3. Funtena? ✓


Gqrx
Gqrx 2.5.3 - addr=192.168.10.2,name,serial=30D2FC1,type=usrp2,uhd
15.321 000 MHz
-50
-63
-76
-89
-102
-115
-128
-141
-154
15.335
15.349
15.363
15.377
15.390
15.404
15.418
15.432
15.446
15.460
15.474
15.488
15.502
15.515
15.529
15.540
15.557
DELL

Live Demos!



Implications

Implications

How Big is the Problem?

Implications

How Big is the Problem?









Samsung SE310

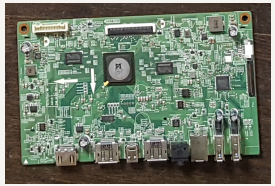
Dell 2417U

Acer G246HL

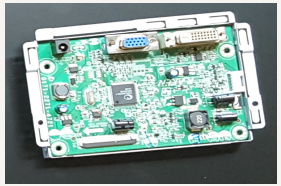
HP 23xw

Implications

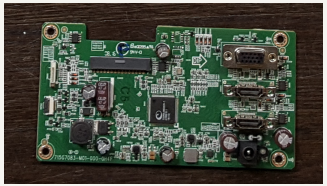
How Big is the Problem?



SE1279RL-MST



MST9122H1



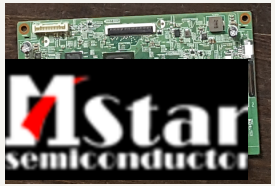
TSUML58YHC2-1



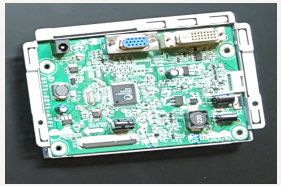
TSUMOL88CDC5-1

Implications

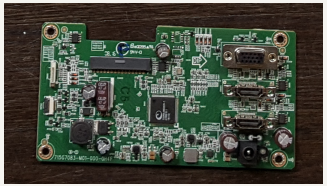
How Big is the Problem?



SE1279RL-MST



MST9122H1



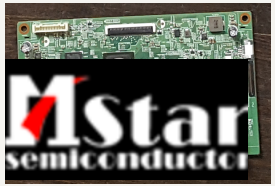
TSUML58YHC2-1



TSUMOL88CDC5-1

Implications

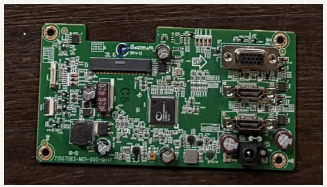
How Big is the Problem?



SE1279RL-MST



MST9122H1



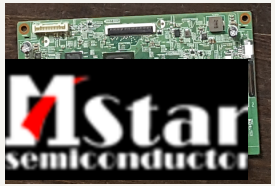
TSUML58YHC2-1



TSUMOL88CDC5-1

Implications

How Big is the Problem?



SE1279RL-MST



MST9122H1



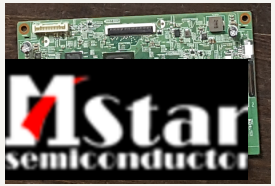
TSUML58YHC2-1



TSUMOL88CDC5-1

Implications

How Big is the Problem?



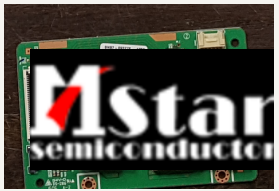
SE1279RL-MST



MST9122H1



TSUML58YHC2-1



TSUMOL88CDC5-1

Implications

How Big is the Problem?



Flashing a BenQ Z-series for free(dom)

2014-04-28, 00:18 Tech and Hacks

DISCLAIMER: If you attempt to reflash your screen's firmwa
that you end up with a very expensive brick. As always, I ca

Alexandre Boeglin

http://boeglin.org/blog/index.php?entry=Flashing-a-BenQ-Z-series-for-free(dom)

Implications

How Big is the Problem?

How Practical is this attack vector?

Implications

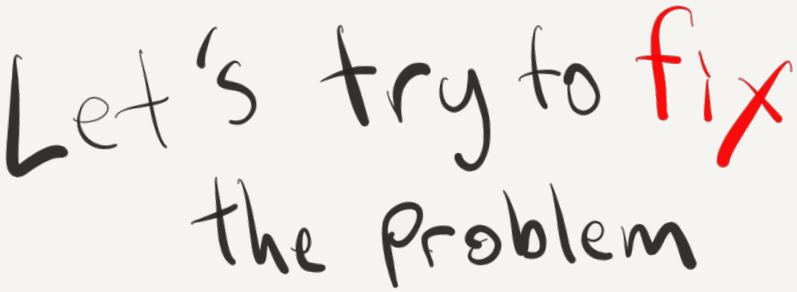
How Big is the Problem?

How Practical is this attack vector?

How Realistic is the fix?

https://github.com/RedBalloonShenanigans/MonitorDarkly


please Contribute!


Let's try to fix
the problem

Conclusion

Work with Cool Kids

Build Embedded Security Tech

Protect All The Things

Jobs@redballoonsecurity.com

Big Thanks!

Artist



BoB Lumsden

Amazing Interns!



Connor Abbd



Brian Hong



DELL DOES NOT YET HAVE A SECURITY FIX AGAINST SHAK ATTACK



MANY MONITORS WERE HARMED IN THE MAKING OF THIS PRESENTATION

arkly



Chris lives happily with his Semi-unmodified 34" monitor