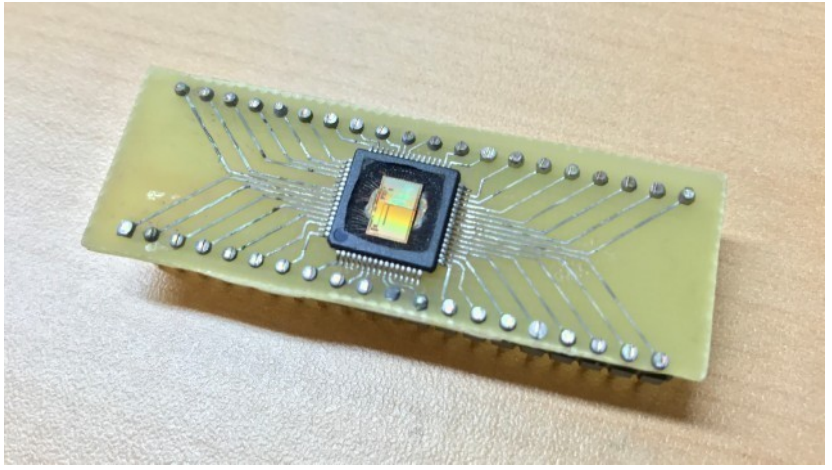
Breaking Code Read Protection on the NXP LPC-family Microcontrollers



Chris Gerlinsky
@akacastor
chris@pdrnorth.com

LPC Microcontrollers

IoT MCUs Based on ARM® Technology

LPC 32-bit ARM MCUs offer exceptional ease of use, design flexibility and advanced integration. LPC continues to transform the MCU landscape with its next-generation LPC800 and LPC54000 series focused on addressing today's IoT design challenges.

LPC800 Series MCUs

An entry-level, 8-bit alternative MCU at the right price

LPC54000 Series MCUs

A power-efficient, mainstream series for everyone

Source:
http://www.nxp.com/products/microcontrollers-and-processors/arm-processors/lpc-cortex-m-mcus

LPC Series MCUs

	CPU	Memory	Features	Applications
LPC800 › Entry-level, 8-bit alternate	ARM® Cortex® - M0+ 30 MHz	8-32 KB Flash 1-8 KB RAM	SCTimer/PWM Switch matrix Pattern match engine	Sensor gateways Battery-powered devices 8/16-bit replacements
LPC54000 › Mainstream, power efficiency	Cortex-M4* 100 MHz 180 MHz *Dual-core options available	128-512 KB Flash Up to 200 KB RAM	USB Digital mic interface CAN/CAN-FD LCD Ethernet FlexComm Security	Smart home and building automation Auto-aftermarket w/ CAN FD Industrial control Gaming accessories
LPC1100 › Entry level, integrated connectivity	Cortex-M0/M0+ 50 MHz	4-256 KB Flash 1-36 KB RAM	USB CAN EEPROM	Gaming and PC peripherals Remote sensors System supervisors Alarm/lighting systems
LPC1200 › Noise immunity	Cortex-M0 50 MHz	32-128 KB Flash 4-8 KB RAM	8 kV protection IEC 60730 Class B certified	White goods Industrial control UPS/power conversion
LPC1300 › Entry-level upgrade	Cortex-M3 72 MHz	8-64 KB Flash 4-12 KB RAM	USB	Consumer peripherals and toys Home automation
LPC1500 › Motion control	Cortex-M3 72 MHz	64-256 KB Flash 12-36 KB RAM	USB CAN QEI	Motor control Digital power supplies Solar inverters
LPC1700 › Scalable, mainstream	Cortex-M3 Up to 120 MHz	32-512 KB Flash 8-96 KB RAM	USB CAN Ethernet LCD QEI	Smart energy Data collectors Auto-aftermarket Industrial controls and networking Medical diagnostics
LPC1800 › Performance and integration	Cortex-M3 180 MHz	Up to 1024 KB Flash* 104-200 KB RAM *Flashless option available	Dual HS USB CAN Ethernet LCD Security	Secure industrial gateways Data collectors Portable medical equipment Consumer audio applications
LPC4000 › Scalable, mainstream	Cortex-M4 120 MHz	64-512 KB Flash 24-96 KB RAM	USB CAN Ethernet LCD	Rich display HMI Medical diagnostics HVAC and building control
LPC4300 › High performance and integration	Cortex-M4* 204 MHz *Multi-core options available	Up to 1024 KB Flash* 104-282 KB RAM *Flashless option available	Dual HS USB CAN Ethernet LCD SGPIO Security	High-fidelity embedded audio Secure communication hubs Data collectors Scanners and printers
LPC2000 › Legacy mainstream microcontrollers	ARM7TDMI-S Up to 72 MHz	8-512 KB Flash 2-128 KB RAM	USB CAN Ethernet LCD	Legacy MCUs for general embedded applications
LPC3000 › Legacy application processors	ARM9 Up to 270 MHz	Flashless Up to 192 KB RAM	USB LCD Ethernet Vector floating point co-processor	Legacy MPUs for general embedded applications

21.13 ISP commands

The following commands are accepted by the ISP command handler. Detailed status codes are supported for each command. The command handler sends the return code INVALID_COMMAND when an undefined command is received. Commands and return codes are in ASCII format.

CMD_SUCCESS is sent by ISP command handler only when received ISP command has been completely executed and the new ISP command can be given by the host. Exceptions from this rule are "Set Baud Rate", "Write to RAM", "Read Memory", and "Go" commands.

Table 317. ISP command summary

ISP Command	Usage	Described in
Unlock	U <Unlock Code>	Table 318
Set Baud Rate	B <Baud Rate> <stop bit>	Table 319
Echo	A <setting>	Table 320
Write to RAM	W <start address> <number of bytes>	Table 321
Read Memory	R <address> <number of bytes>	Table 322
Prepare sector(s) for write operation	P <start sector number> <end sector number>	Table 323
Copy RAM to flash	C <Flash address> <RAM address> <number of bytes>	Table 324
Go	G <address> <Mode>	Table 325
Erase sector(s)	E <start sector number> <end sector number>	Table 326
Blank check sector(s)	I <start sector number> <end sector number>	Table 327
Read Part ID	J	Table 328
Read Boot code version	K	Table 330
Compare	M <address1> <address2> <number of bytes>	Table 331
ReadUID	N	Table 332

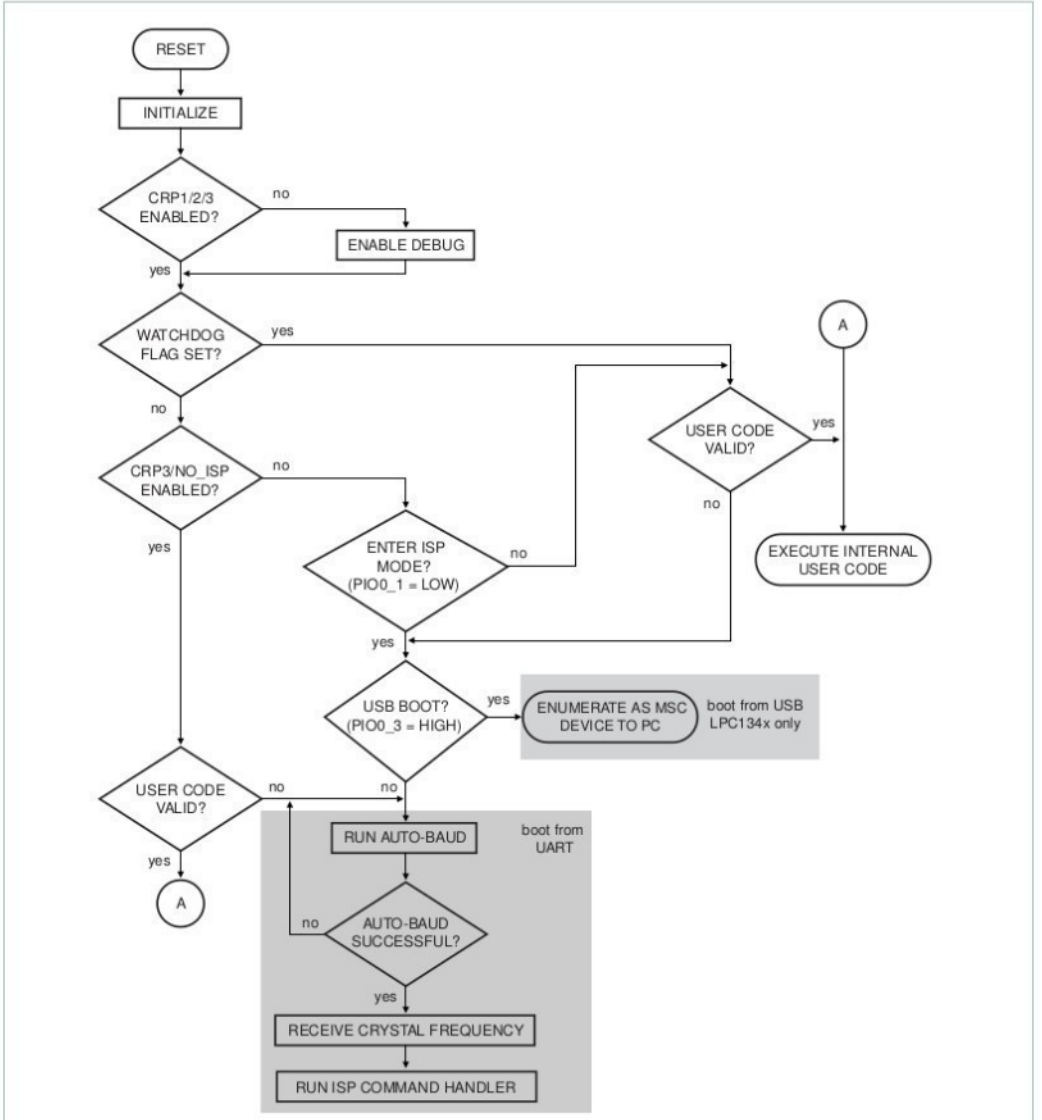
Source: LPC1343 User Manual

Table 314. Code Read Protection (CRP) options

Name	Pattern programmed in 0x0000 02FC	Description
NO_ISP	0x4E69 7370	Prevents sampling of pin PIO0_1 for entering ISP mode. PIO0_1 is available for other uses.
CRP1	0x12345678	Access to chip via the SWD pins is disabled. This mode allows partial flash update using the following ISP commands and restrictions: • Write to RAM command cannot access RAM below 0x1000 0300. • Copy RAM to flash command can not write to Sector 0. • Erase command can erase Sector 0 only when all sectors are selected for erase. • Compare command is disabled. • Read Memory command is disabled. This mode is useful when CRP is required and flash field updates are needed but all sectors can not be erased. Since compare command is disabled in case of partial updates the secondary loader should implement checksum mechanism to verify the integrity of the flash.
CRP2	0x87654321	Access to chip via the SWD pins is disabled. The following ISP commands are disabled: • Read Memory • Write to RAM • Go • Copy RAM to flash • Compare When CRP2 is enabled the ISP erase command only allows erasure of all user sectors.
CRP3	0x43218765	Access to chip via the SWD pins is disabled. ISP entry by pulling PIO0_1 LOW is disabled if a valid user code is present in flash sector 0. This mode effectively disables ISP override using PIO0_1 pin. It is up to the user's application to provide a flash update mechanism using IAP calls or call reinvoke ISP command to enable flash update via UART0. **Caution: If CRP3 is selected, no future factory testing can be performed on the device.**

Source: LPC1343 User Manual

21.10 Boot process flowchart



Source: LPC1343 User Manual

Values at FLASH:02FC that **enable** Code Read Protection:

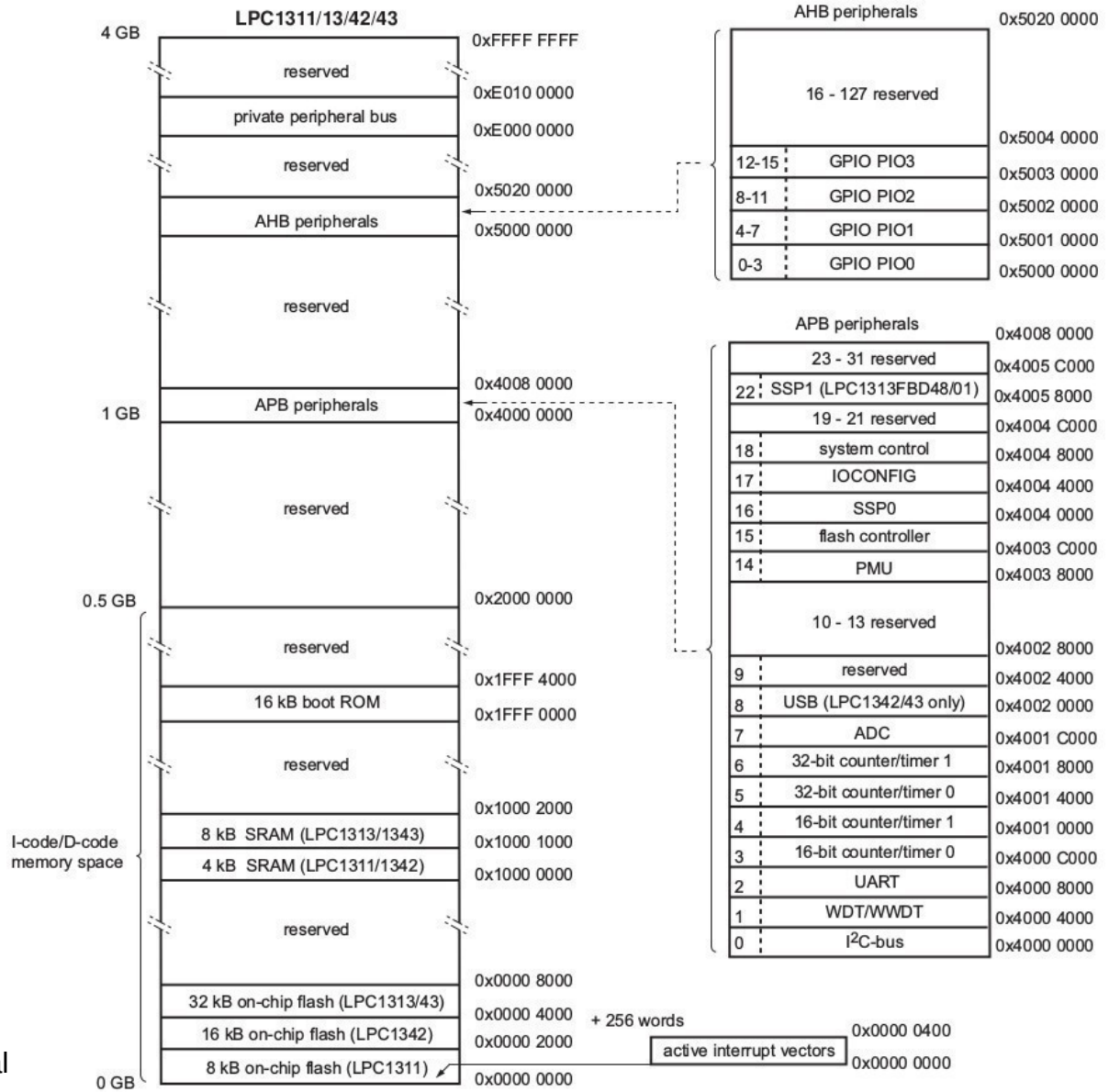
```
0x12345678  CRP1
0x87654321  CRP2
0x43218765  CRP3
0x4E697370  NO_ISP
```

4 possible 32-bit words

Values at FLASH:02FC that **disable** Code Read Protection:

```
0xFFFFFFFF
0x00000000
0xABCDEF12
0x12345679
0x87654320
0x53218765
0x4E687370
...
```

4,294,967,292 possible 32-bit words



Source: LPC1343 User Manual

Useful tools and links:

lpc21isp – software to program LPC-family microcontrollers using ISP

http://eleceng.dit.ie/frank/arm/BareMetalLPC812/index.html
http://eleceng.dit.ie/frank/arm/BareMetalLPC812/serial.tar.gz

http://eleceng.dit.ie/frank/arm/BareMetalLPC1114/index.html

git://gnudd.com/cortex-m3.git http://www.gnudd.com/wd/cortex-m3.pdf

https://www.olimex.com/Products/ARM/NXP/LPC-P2148/

http://siwawi.bauing.uni-kl.de/avr_projects/arm_projects/lpc2k_bundle_port/lpc213x_lpc214x_examples_20061205.zip

```c
void dump_bootrom(void)
{
    unsigned char *FlashAddr;


    // loop through boot ROM 1FFF0000 - 1FFFFFFF and send it as ASCII hex bytes
    for( FlashAddr=(unsigned char *)0x1FFF0000; FlashAddr<=(unsigned char *)0x1FFFFFFF; FlashAddr++ )
        printf( "%02X ", *FlashAddr );
}
```

```
ROM:1FFF010C
ROM:1FFF010C                boot_Startup                                    ; CODE XREF: ROM:1FFF0106↑j
ROM:1FFF010C                                                                ; DATA XREF: ROM:loc_1FFF0104↑o
ROM:1FFF010C                                                                ; ROM:off_1FFF0108↑o
ROM:1FFF010C 1A 4A                      LDR     R2, =dword_400483F0
ROM:1FFF010E 1B 4B                      LDR     R3, =addr_of_FLASH_CRP  ; FLASH:02FC is where CRP value is located in application flash
ROM:1FFF0110 1B 68                      LDR     R3, [R3] ; FLASH_CRP_location ; FLASH:02FC is where CRP value is located in application flash
ROM:1FFF0112 1C 4D                      LDR     R5, =CRP3_value         ; CRP3 0x43218765
ROM:1FFF0114 2D 68                      LDR     R5, [R5]                ; CRP3 0x43218765
ROM:1FFF0116 1C 4E                      LDR     R6, =CRP1_value         ; CRP1 0x12345678
ROM:1FFF0118 36 68                      LDR     R6, [R6]                ; CRP1 0x12345678
ROM:1FFF011A 1C 68                      LDR     R4, [R3]                ; R4 contains CRP word from FLASH:02FC
ROM:1FFF011C AC 42                      CMP     R4, R5
ROM:1FFF011E 01 D0                      BEQ     in_mode_CRP1_or_CRP3
ROM:1FFF0120 B4 42                      CMP     R4, R6
ROM:1FFF0122 01 D1                      BNE     not_CRP1
ROM:1FFF0124
ROM:1FFF0124                in_mode_CRP1_or_CRP3                            ; CODE XREF: boot_Startup+12↑j
ROM:1FFF0124 16 4C                      LDR     R4, =CRP2_value         ; in modes CRP1 or CRP3, load R4 with same value as CRP2 (0x87654321)
ROM:1FFF0126 24 68                      LDR     R4, [R4]                ; CRP2 0x87654321
ROM:1FFF0128
ROM:1FFF0128                not_CRP1                                        ; CODE XREF: boot_Startup+16↑j
ROM:1FFF0128 14 60                      STR     R4, [R2]                ; store CRP2 value @ 400483F0 to disable debug mode
ROM:1FFF012A 0D 4A                      LDR     R2, =FMC_4003C000       ; FMC flash controller base address
ROM:1FFF012C 13 68                      LDR     R3, [R2]                ; FMC flash controller base address
ROM:1FFF012E 40 24                      MOVS    R4, #0x40 ; '@'
ROM:1FFF0130 23 43                      ORRS    R3, R4
ROM:1FFF0132 13 60                      STR     R3, [R2]                ; FMC flash controller base address
```

```
ROM:1FFF12E6
ROM:1FFF12E6 F8 B5                      PUSH            {R3-R7,LR}
ROM:1FFF12E8 43 4A                      LDR             R2, =FMC_4003C000       ; FMC flash controller base address
ROM:1FFF12EA 10 68                      LDR             R0, [R2]
ROM:1FFF12EC 40 23                      MOVS            R3, #0x40 ; '@'
ROM:1FFF12EE 98 43                      BICS            R0, R3                  ; clear bit 0x40 of FMC_4003C000 to enable access to user flash area
ROM:1FFF12F0 10 60                      STR             R0, [R2]
ROM:1FFF12F2 43 48                      LDR             R0, =addr_of_FLASH_CRP  ; FLASH:02FC is where CRP value is located in application flash
ROM:1FFF12F4 33 49                      LDR             R1, =dword_1000017C     ; this + 8 = CRP_value_in_RAM
ROM:1FFF12F6 00 68                      LDR             R0, [R0] ; FLASH_CRP_location ; FLASH:02FC is where CRP value is located in application flash
ROM:1FFF12F8 00 68                      LDR             R0, [R0]
ROM:1FFF12FA 88 60                      STR             R0, [R1,#(CRP_value_in_RAM - 0x1000017C)]
```

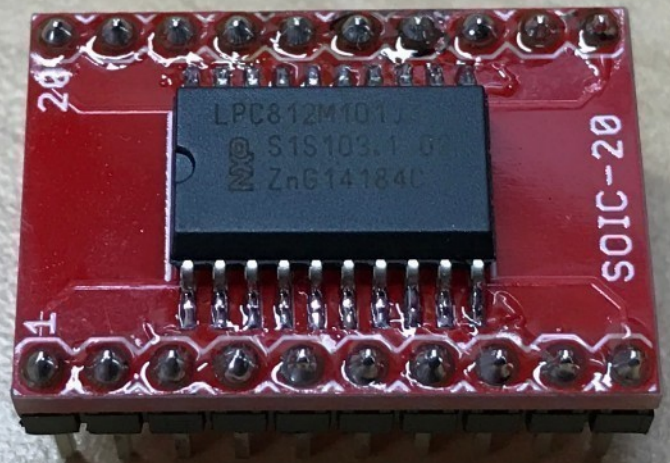
```
ROM:1FFF1396
ROM:1FFF1396                loc_1FFF1396                                        ; CODE XREF: boot_continue_startup+AA↑j
ROM:1FFF1396 38 6B                      LDR             R0, [R7,#0x30]
ROM:1FFF1398 40 07                      LSLS            R0, R0, #29
ROM:1FFF139A 54 D4                      BMI             to_run_flash_app_or_bootloader ; branch if bit 2 of SYSTICK is set ?
ROM:1FFF139C 19 48                      LDR             R0, =unk_50003FC0       ; GPIODATAMASK
ROM:1FFF139E C0 6B                      LDR             R0, [R0,#0x3C]          ; R0 = GPIODATA
ROM:1FFF13A0 80 07                      LSLS            R0, R0, #30
ROM:1FFF13A2 50 D4                      BMI             to_run_flash_app_or_bootloader ; branch if bit GPIO0.1 is set (P0.1 not held low for bootloader
ROM:1FFF13A4 07 48                      LDR             R0, =dword_1000017C     ; this + 8 = CRP_value_in_RAM
ROM:1FFF13A6 05 49                      LDR             R1, =CRP3_value         ; CRP3 0x43218765
ROM:1FFF13A8 80 68                      LDR             R0, [R0,#(CRP_value_in_RAM - 0x1000017C)]
ROM:1FFF13AA 09 68                      LDR             R1, [R1]                ; CRP3 0x43218765
ROM:1FFF13AC 88 42                      CMP             R0, R1
ROM:1FFF13AE 4A D0                      BEQ             to_run_flash_app_or_bootloader
ROM:1FFF13B0 22 49                      LDR             R1, =0x4E697370         ; CRP NO_ISP value
ROM:1FFF13B2 88 42                      CMP             R0, R1
ROM:1FFF13B4 47 D0                      BEQ             to_run_flash_app_or_bootloader
ROM:1FFF13B6 43 E0                      B               to_select_bootloader
ROM:1FFF13B6                ; ---------------------------------------------------------------------------
```

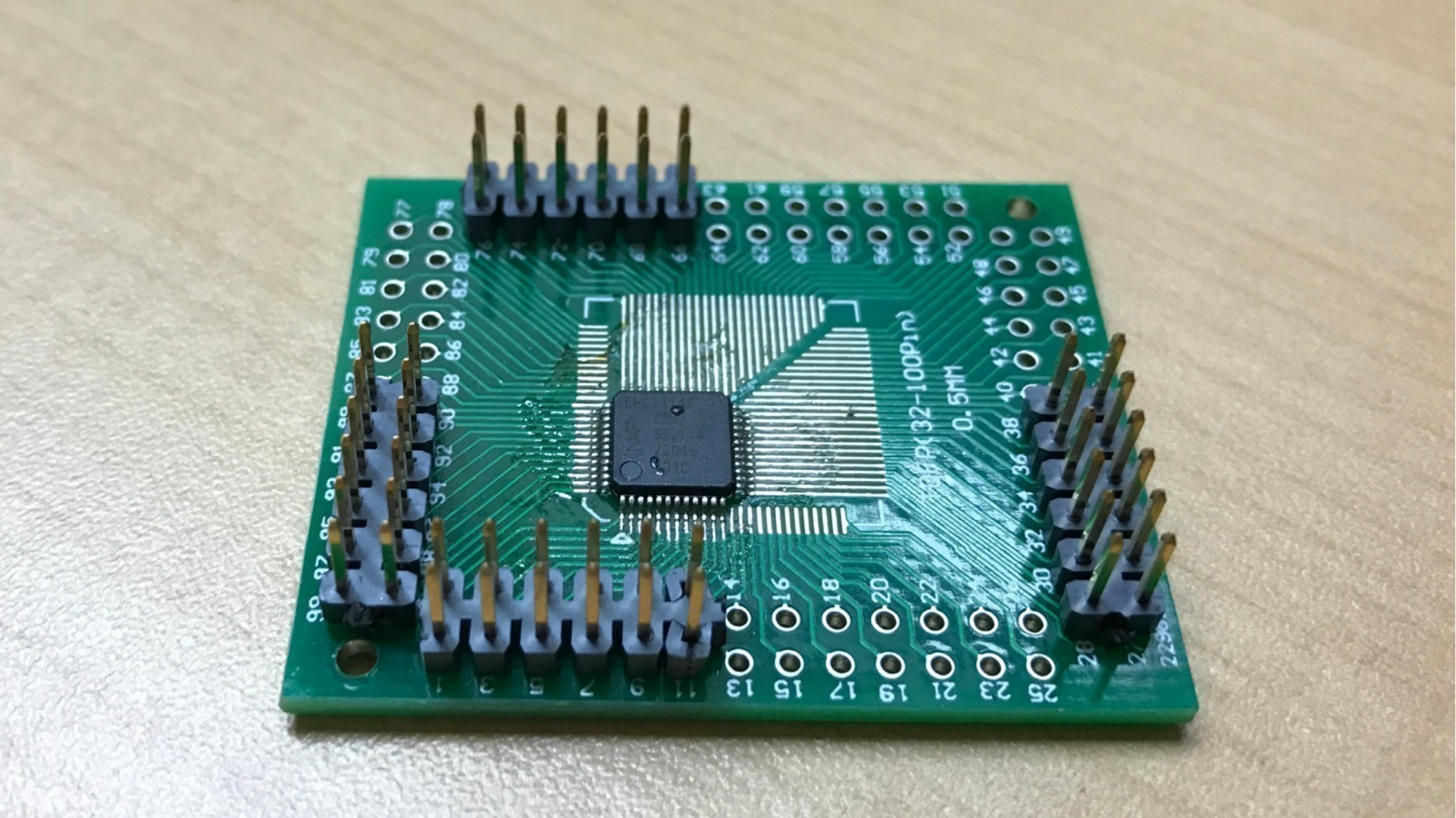
```
ROM:1FFF12DA                run_flash_app_or_bootloader                         ; CODE XREF: boot_continue_startup:to_run_flash_app_or_bootloader↓p
ROM:1FFF12DA 10 B5                      PUSH            {R4,LR}
ROM:1FFF12DC FF F7 DD FF                BL              run_flash_app_if_valid  ; starts user application in FLASH if it is valid
ROM:1FFF12E0 FF F7 BB FF                BL              select_bootloader
ROM:1FFF12E0                ; End of function run_flash_app_or_bootloader
ROM:1FFF12E0
```

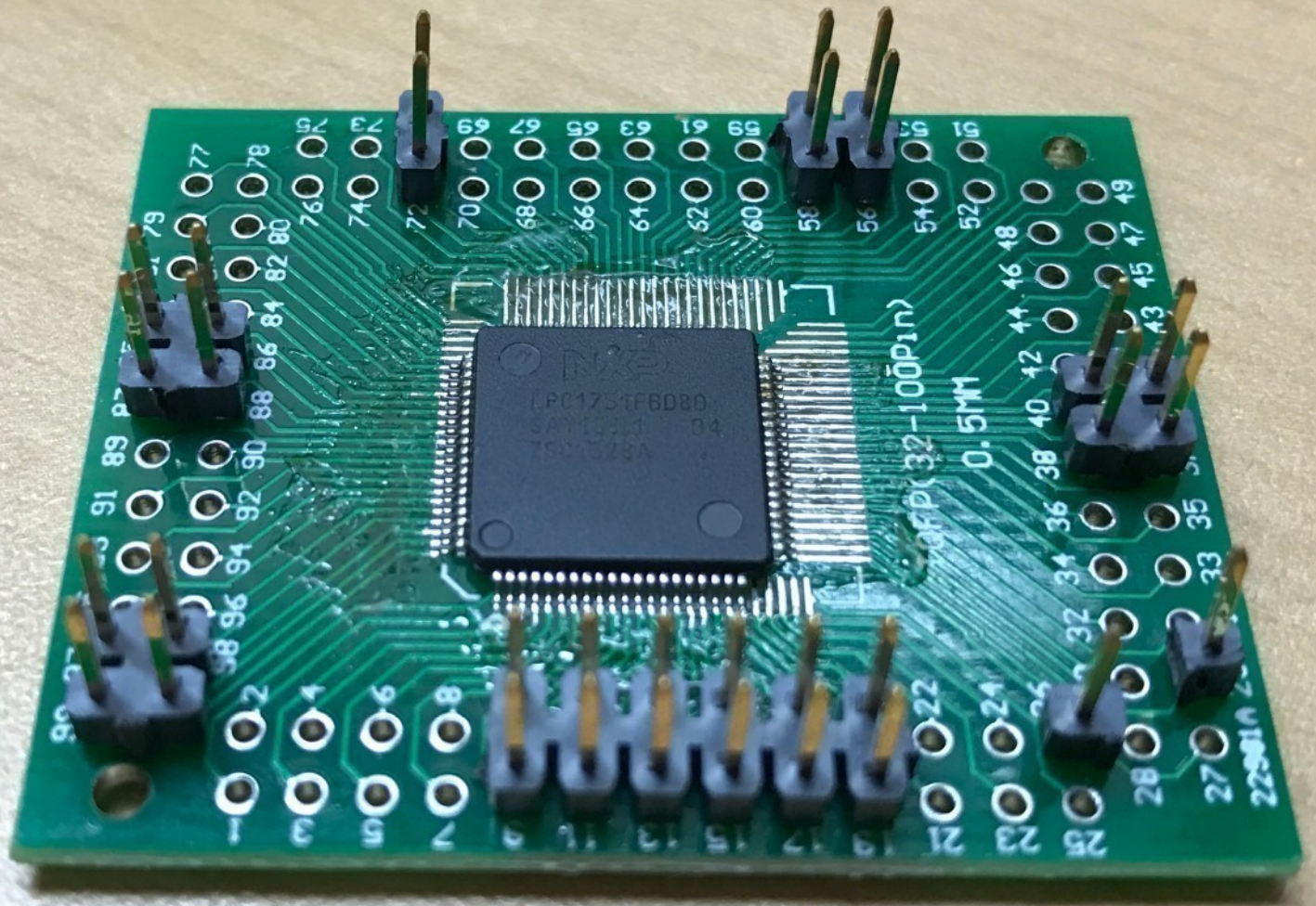
```
ROM:1FFF125A                select_bootloader                                       ; CODE XREF: sub_1FFF01A4+8↑p
ROM:1FFF125A                                                                        ; run_flash_app_or_bootloader+6↓p
ROM:1FFF125A                                                                        ; boot_continue_startup:to_select_bootloader↓p
ROM:1FFF125A 10 B5                      PUSH            {R4,LR}
ROM:1FFF125C 66 48                      LDR             R0, =FMC_4003C000       ; FMC flash controller base address
ROM:1FFF125E 02 68                      LDR             R2, [R0]
ROM:1FFF1260 40 21                      MOVS            R1, #0x40 ; '@'
ROM:1FFF1262 0A 43                      ORRS            R2, R1
ROM:1FFF1264 02 60                      STR             R2, [R0]
ROM:1FFF1266 65 4A                      LDR             R2, =dword_430
ROM:1FFF1268 14 68                      LDR             R4, [R2]
ROM:1FFF126A 02 68                      LDR             R2, [R0]
ROM:1FFF126C 8A 43                      BICS            R2, R1
ROM:1FFF126E 02 60                      STR             R2, [R0]
ROM:1FFF1270 63 48                      LDR             R0, =addr_of_FLASH_CRP  ; FLASH:02FC is where CRP value is located in application flash
ROM:1FFF1272 54 49                      LDR             R1, =dword_1000017C     ; this + 8 = CRP_value_in_RAM
ROM:1FFF1274 00 68                      LDR             R0, [R0] ; FLASH_CRP_location ; FLASH:02FC is where CRP value is located in application flash
ROM:1FFF1276 00 68                      LDR             R0, [R0]
ROM:1FFF1278 88 60                      STR             R0, [R1,#(CRP_value_in_RAM - 0x1000017C)]
ROM:1FFF127A FF F7 D4 FF                BL              sub_1FFF1226            ; some CLK setup (?)
ROM:1FFF127E E0 07                      LSLS            R0, R4, #31
ROM:1FFF1280 08 D1                      BNE             to_enter_serial_ISP
ROM:1FFF1282 60 48                      LDR             R0, =unk_50003FC0
ROM:1FFF1284 C0 6B                      LDR             R0, [R0,#0x3C]          ; R0 = GPIODATA
ROM:1FFF1286 00 07                      LSLS            R0, R0, #28             ; check P0.3 / USB_VBUS
ROM:1FFF1288 04 D5                      BPL             to_enter_serial_ISP     ; if P0.3 / USB_VBUS is low then enter serial ISP
ROM:1FFF1288                                                                    ; (high would indicate USB bootloader should be started)
ROM:1FFF128A FF F7 86 FF                BL              sub_1FFF119A
ROM:1FFF128E 01 F0 66 F8                BL              sub_1FFF235E
ROM:1FFF1292                ; ---------------------------------------------------------------------------
ROM:1FFF1292 10 BD                      POP             {R4,PC}
ROM:1FFF1294                ; ---------------------------------------------------------------------------
ROM:1FFF1294
ROM:1FFF1294                to_enter_serial_ISP                                     ; CODE XREF: select_bootloader+26↑j
ROM:1FFF1294                                                                        ; select_bootloader+2E↑j
ROM:1FFF1294 FF F7 4C FF                BL              enter_serial_ISP
ROM:1FFF1298                ; ---------------------------------------------------------------------------
```

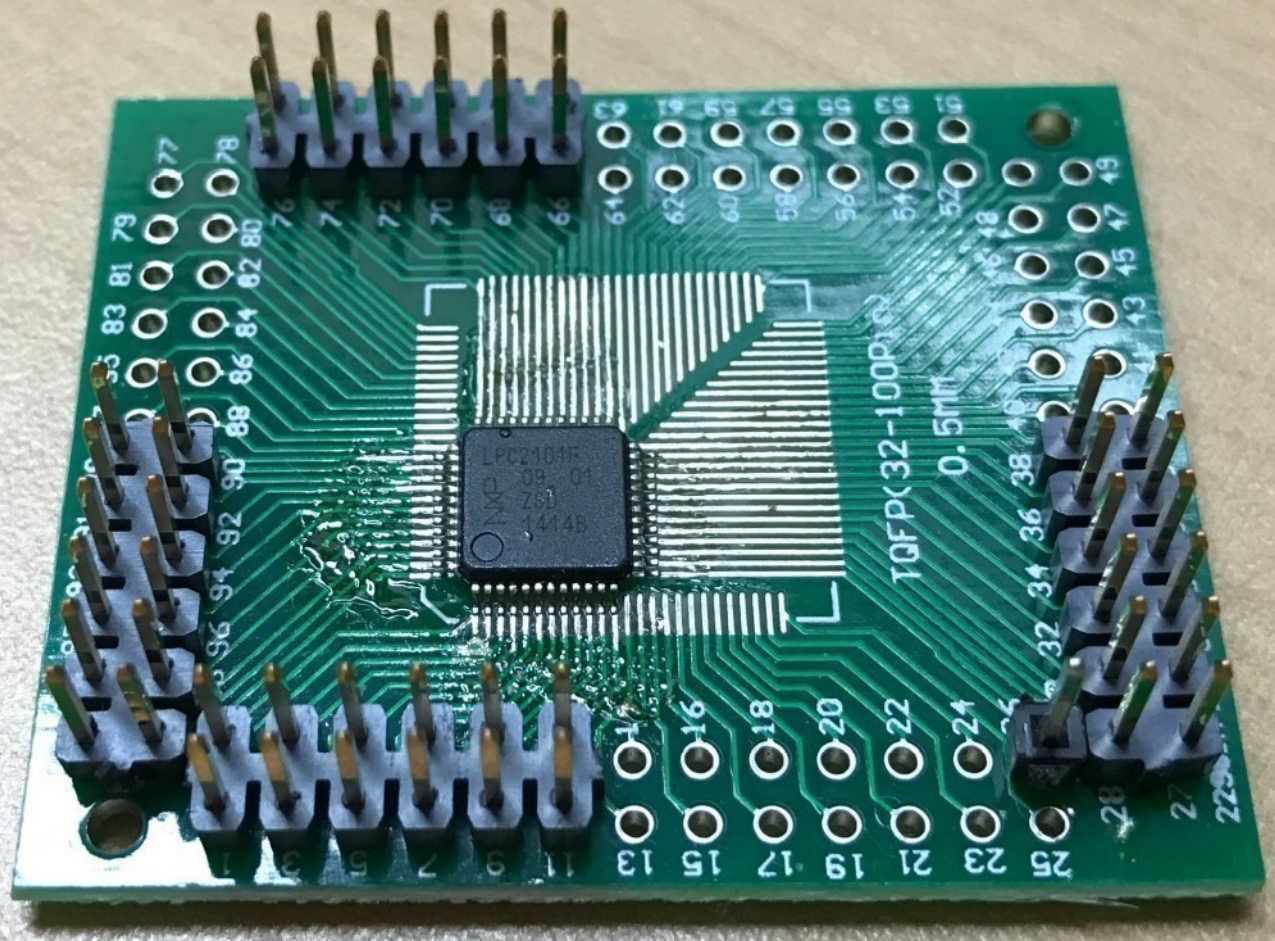
```
ROM:1FFF0FBC                serial_ISP                                          ; CODE XREF: enter_serial_ISP+4A↓p
ROM:1FFF0FBC
ROM:1FFF0FBC                var_20          = -0x20
ROM:1FFF0FBC                var_1C          = -0x1C
ROM:1FFF0FBC
ROM:1FFF0FBC FE B5                          PUSH            {R1-R7,LR}
ROM:1FFF0FBE FE 4E                          LDR             R6, =CRP2_value     ; CRP2 0x87654321
ROM:1FFF0FC0 FE 4F                          LDR             R7, =CRP3_value     ; CRP3 0x43218765
ROM:1FFF0FC2 FF 4D                          LDR             R5, =dword_10000248
ROM:1FFF0FC4
ROM:1FFF0FC4                isp_command_loop                                    ; CODE XREF: serial_ISP+16↓j
ROM:1FFF0FC4                                                                    ; serial_ISP+3A↓j
ROM:1FFF0FC4                                                                    ; serial_ISP+9E↓j
ROM:1FFF0FC4                                                                    ; serial_ISP+A4↓j
ROM:1FFF0FC4 FE 48                          LDR             R0, =dword_10000248
ROM:1FFF0FC6 01 AA                          ADD             R2, SP, #0x20+var_1C
ROM:1FFF0FC8 46 21                          MOVS            R1, #0x46 ; 'F'
ROM:1FFF0FCA 94 38                          SUBS            R0, #0x94 ; ''
ROM:1FFF0FCC 00 F0 5B FE                    BL              uart_gets           ; this does serial RX of ASCII string
ROM:1FFF0FD0 00 28                          CMP             R0, #0
ROM:1FFF0FD2 F7 D1                          BNE             isp_command_loop
```

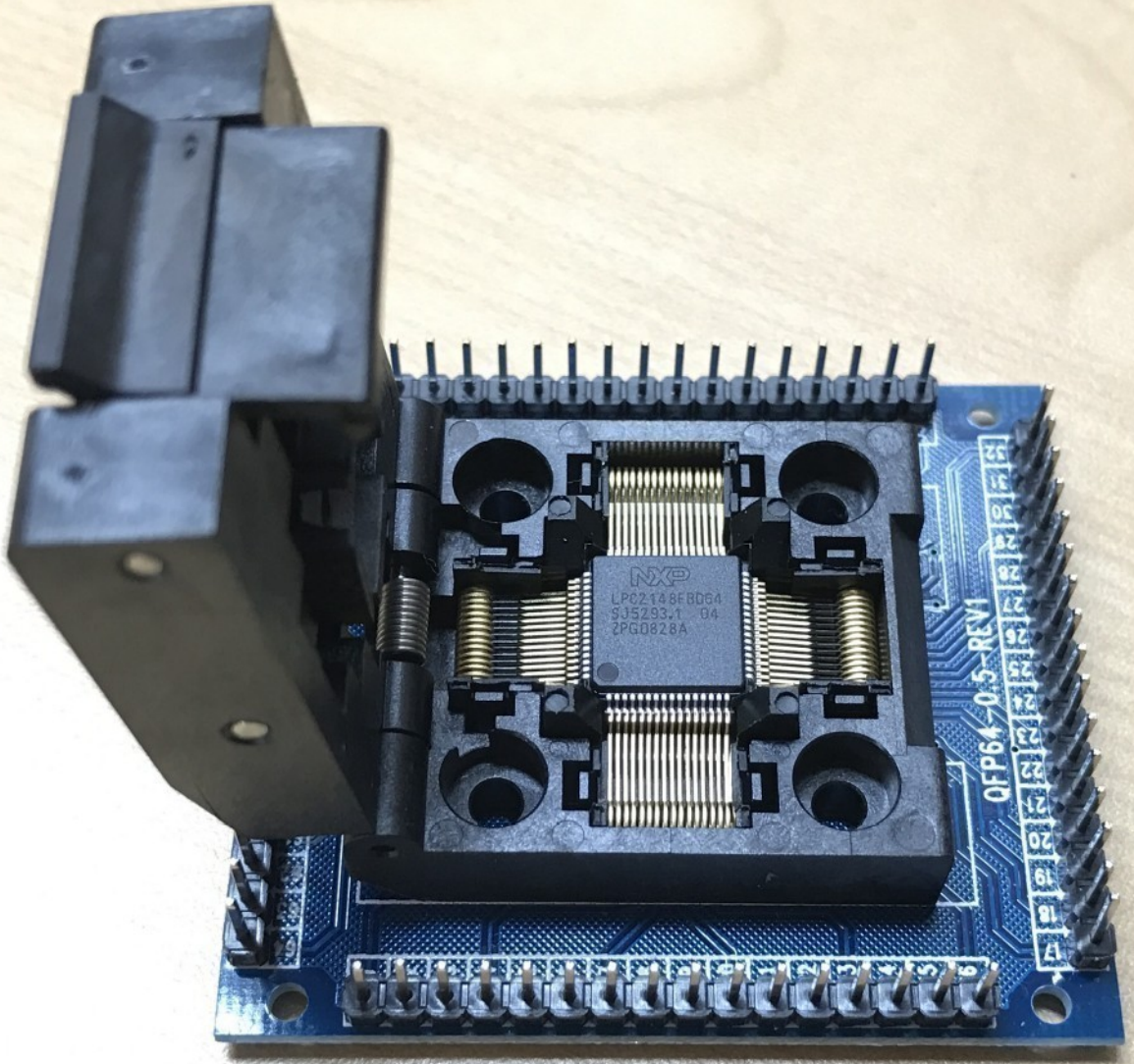
```
ROM:1FFF100C                command_disabled_if_CRP_enabled                     ; CODE XREF: serial_ISP+42↑j
ROM:1FFF100C                                                                    ; serial_ISP+46↑j
ROM:1FFF100C                                                                    ; serial_ISP+4A↑j
ROM:1FFF100C 01 22                  MOVS        R2, #1                          ; R2=1 indicates command not allowed if CRP1, 2, or 3 enabled
ROM:1FFF100E 07 E0                  B           check_current_CRP_level
ROM:1FFF1010                ; ---------------------------------------------------------------------------
ROM:1FFF1010
ROM:1FFF1010                loc_1FFF1010                                        ; CODE XREF: serial_ISP+4E↑j
ROM:1FFF1010 00 22                  MOVS        R2, #0
ROM:1FFF1012 57 29                  CMP         R1, #0x57 ; 'W'                 ; 'W'rite command
ROM:1FFF1014 01 D0                  BEQ         command_disabled_if_CRP2_or_3
ROM:1FFF1016 43 29                  CMP         R1, #0x43 ; 'C'
ROM:1FFF1018 01 D1                  BNE         command_not_disabled_by_CRP
ROM:1FFF101A
ROM:1FFF101A                command_disabled_if_CRP2_or_3                       ; CODE XREF: serial_ISP+58↑j
ROM:1FFF101A 01 24                  MOVS        R4, #1
ROM:1FFF101C 00 E0                  B           check_current_CRP_level
ROM:1FFF101E                ; ---------------------------------------------------------------------------
ROM:1FFF101E
ROM:1FFF101E                command_not_disabled_by_CRP                         ; CODE XREF: serial_ISP+5C↑j
ROM:1FFF101E 00 24                  MOVS        R4, #0                          ; R4=0 and R2=0 indicate that CRP does not affect this command
ROM:1FFF1020
ROM:1FFF1020                check_current_CRP_level                             ; CODE XREF: serial_ISP+52↑j
ROM:1FFF1020                                                                    ; serial_ISP+60↑j
ROM:1FFF1020 E8 49                  LDR         R1, =dword_1000017C             ; this + 8 = CRP_value_in_RAM
ROM:1FFF1022 33 68                  LDR         R3, [R6]                        ; CRP2 0x87654321
ROM:1FFF1024 89 68                  LDR         R1, [R1,#(CRP_value_in_RAM - 0x1000017C)]
ROM:1FFF1026 99 42                  CMP         R1, R3                          ; check if CRP2 is set
ROM:1FFF1028 06 D0                  BEQ         CRP1_2_or_3_is_enabled          ; branch if CRP2 is set
ROM:1FFF102A 3B 68                  LDR         R3, [R7]                        ; CRP3 0x43218765
ROM:1FFF102C 99 42                  CMP         R1, R3                          ; check if CRP3 is set
ROM:1FFF102E 03 D0                  BEQ         CRP1_2_or_3_is_enabled          ; branch if CRP3 is set
ROM:1FFF1030 E5 4B                  LDR         R3, =CRP1_value                 ; CRP1 0x12345678
ROM:1FFF1032 1B 68                  LDR         R3, [R3]                        ; CRP1 0x12345678
ROM:1FFF1034 99 42                  CMP         R1, R3                          ; check if CRP1 is set
ROM:1FFF1036 01 D1                  BNE         no_CRP_is_set
ROM:1FFF1038
ROM:1FFF1038                CRP1_2_or_3_is_enabled                              ; CODE XREF: serial_ISP+6C↑j
ROM:1FFF1038                                                                    ; serial_ISP+72↑j
ROM:1FFF1038 01 2A                  CMP         R2, #1                          ; CRP1 / CRP2 / or CRP3
ROM:1FFF103A 07 D0                  BEQ         disallow_command                ; branch if this command requires elevated privileges (ie: not allowed in CRP1/2/3 modes)
ROM:1FFF103C
ROM:1FFF103C                no_CRP_is_set                                       ; CODE XREF: serial_ISP+7A↑j
ROM:1FFF103C 32 68                  LDR         R2, [R6]                        ; CRP2 0x87654321
ROM:1FFF103E 91 42                  CMP         R1, R2
ROM:1FFF1040 02 D0                  BEQ         CRP2_or_3_is_enabled            ; CRP2 / CRP3
ROM:1FFF1042 3A 68                  LDR         R2, [R7]                        ; CRP3 0x43218765
ROM:1FFF1044 91 42                  CMP         R1, R2
ROM:1FFF1046 09 D1                  BNE         ok_to_execute                   ; CRP is not enabled - allow this command to execute
ROM:1FFF1048
ROM:1FFF1048                CRP2_or_3_is_enabled                                ; CODE XREF: serial_ISP+84↑j
ROM:1FFF1048 01 2C                  CMP         R4, #1                          ; CRP2 / CRP3
ROM:1FFF104A 07 D1                  BNE         ok_to_execute                   ; branch if this command is not disabled by CRP2 / CRP3 modes
ROM:1FFF104C
ROM:1FFF104C                disallow_command                                    ; CODE XREF: serial_ISP+7E↑j
ROM:1FFF104C 0F 22                  MOVS        R2, #0xF
ROM:1FFF104E 13 20                  MOVS        R0, #19                         ; R0 = response code (19 indicates command disabled by CRP)
```

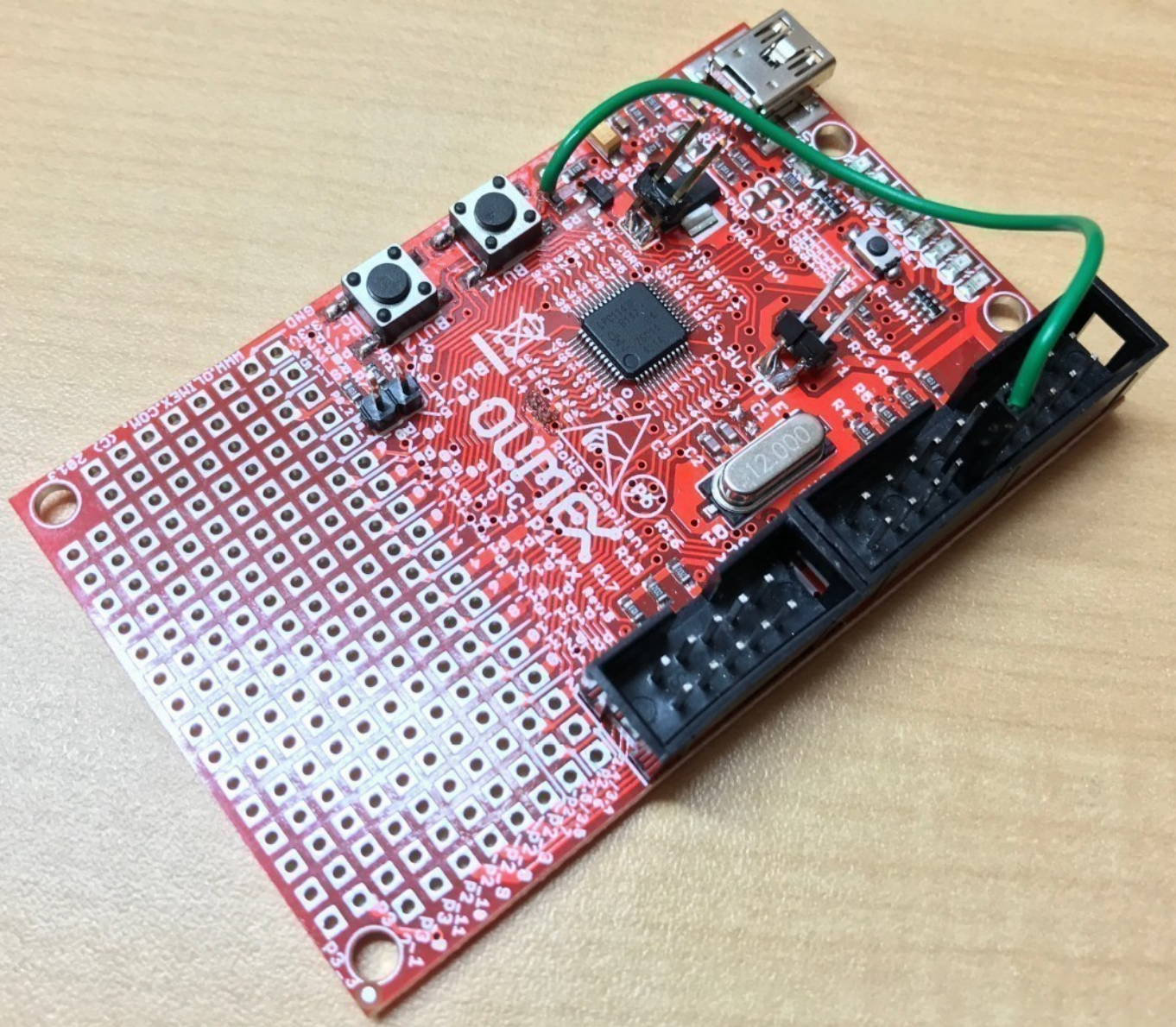

20
1
LPC812M101J
SOIC-20

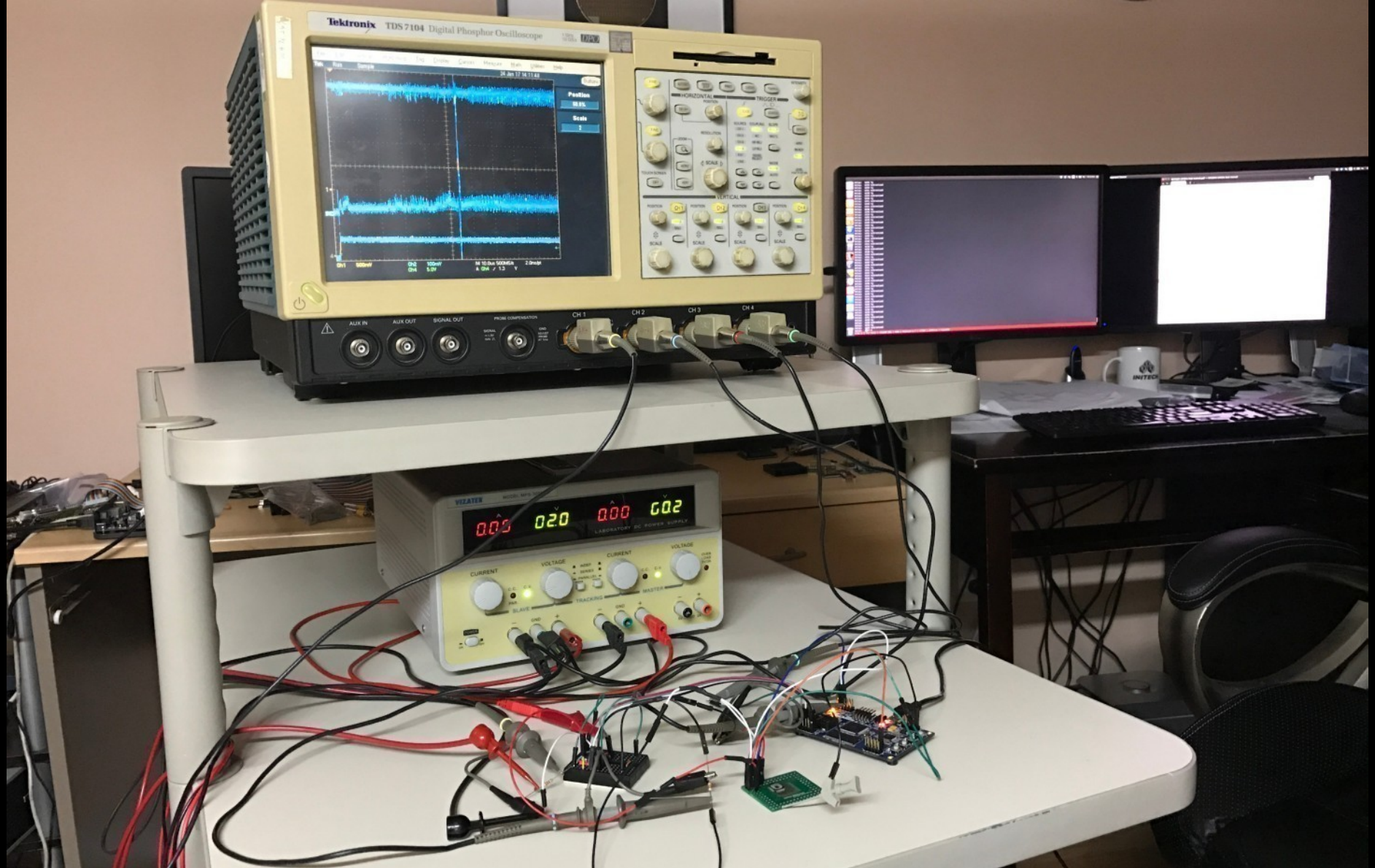

0.5MM


LPC1751FBD80
0.5MM


LPC2101F
TQFP(32-100P)
0.5mm

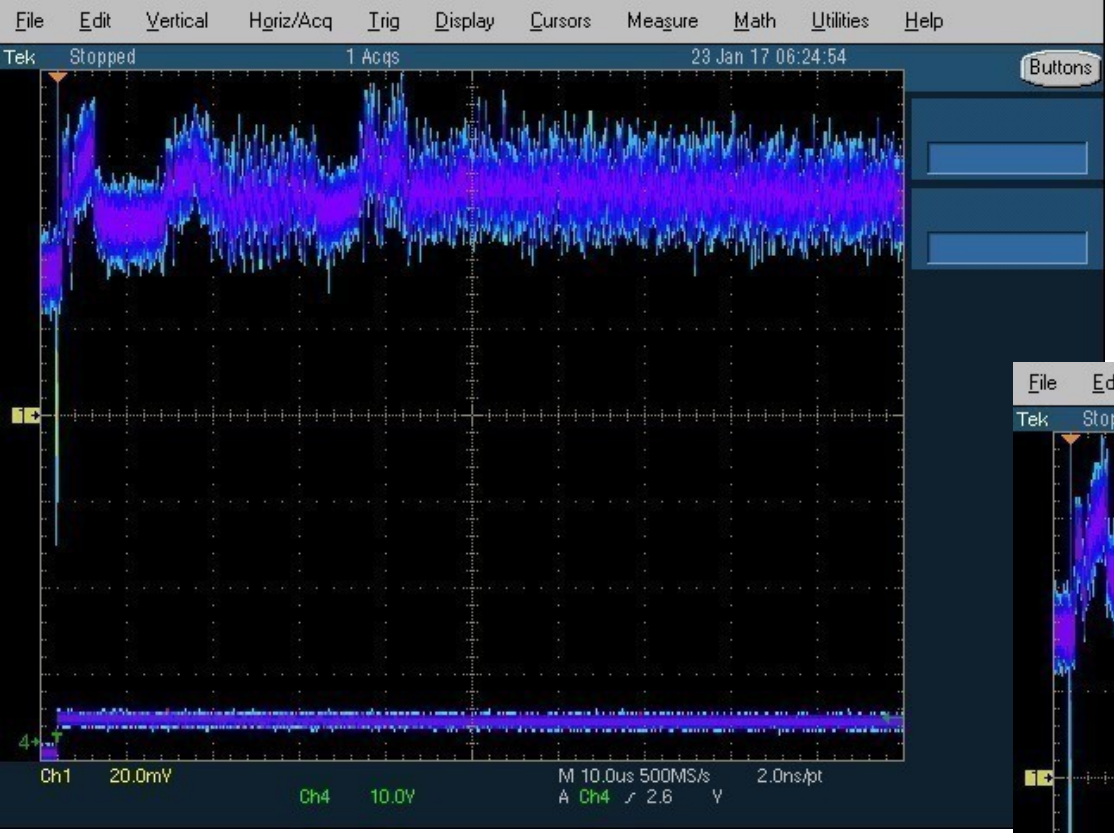

NXP
LPC2148FBD64
SJ5293.1 04
2PGD828A
QFP64-0.5-REV1


12.000


Tektronix TDS 7104 Digital Phosphor Oscilloscope
AUX IN
AUX OUT
SIGNAL OUT
CH 1
CH 2
CH 3
CH 4
VIZATEX
CURRENT
VOLTAGE
TRACKING
MASTER
SLAVE

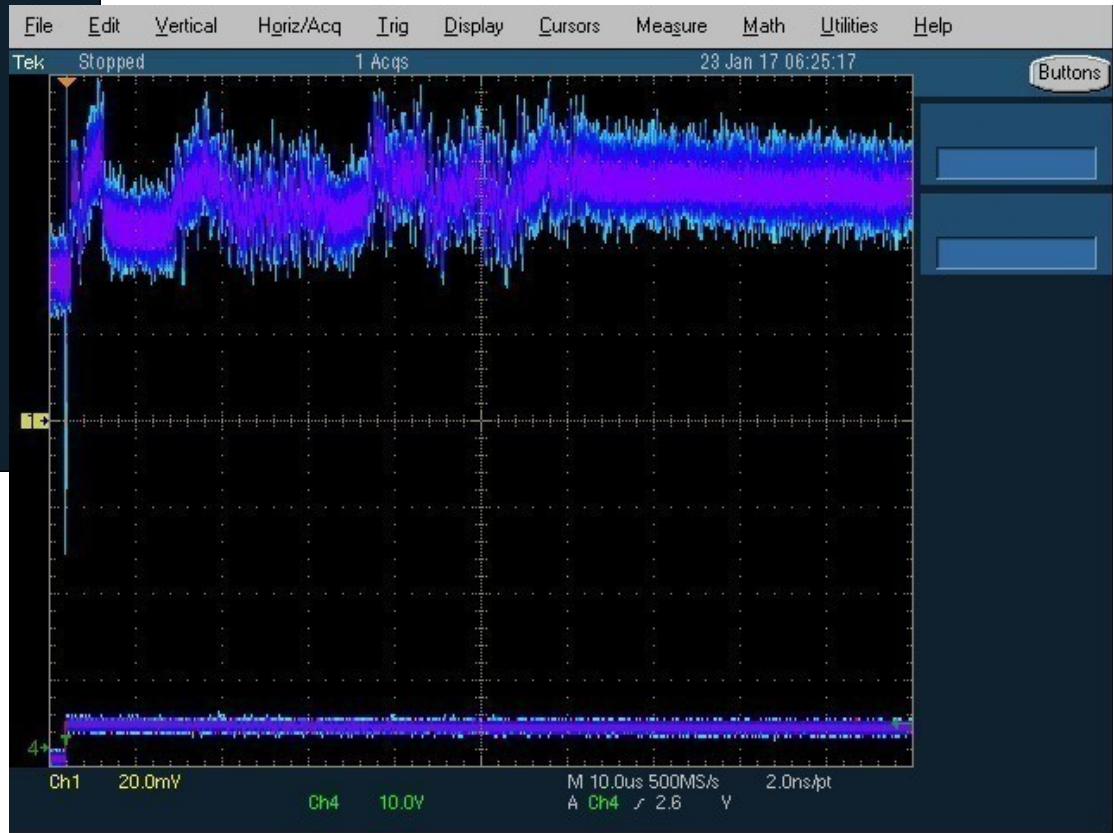
LPC812

Volage measured across 10 ohm resistor
in series with GND



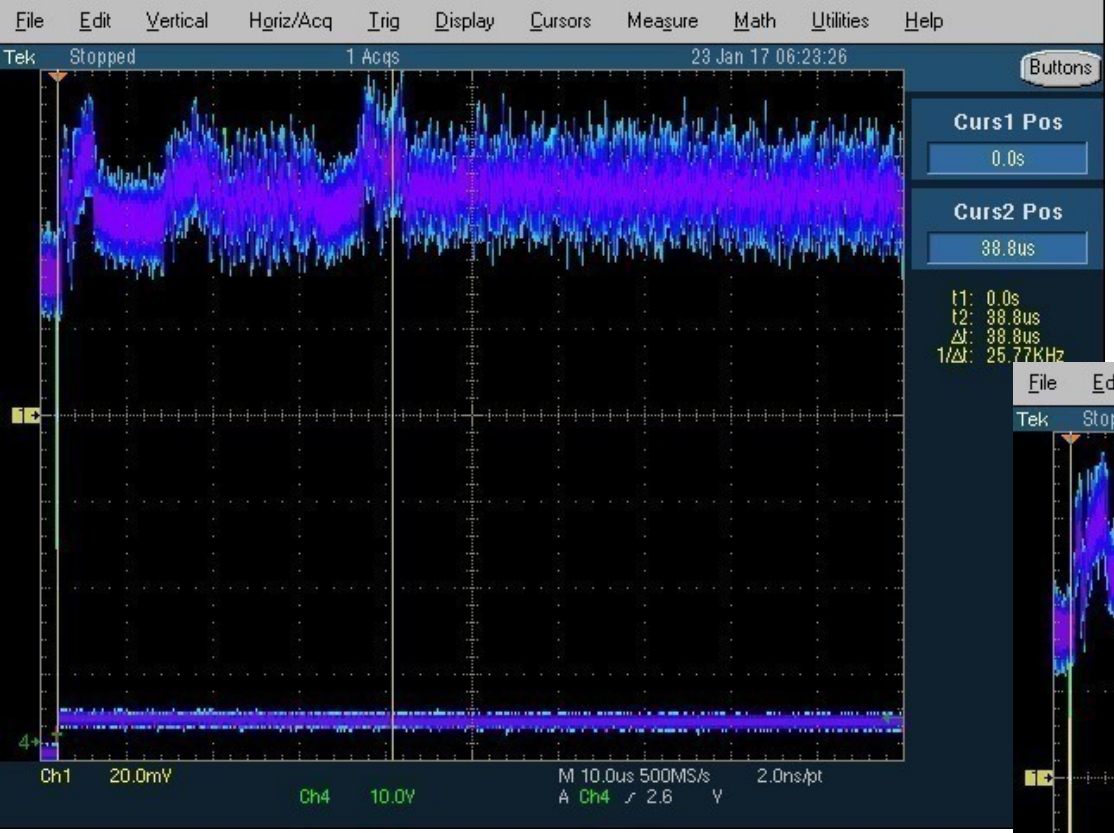
Reset and start application

Reset and start bootloader



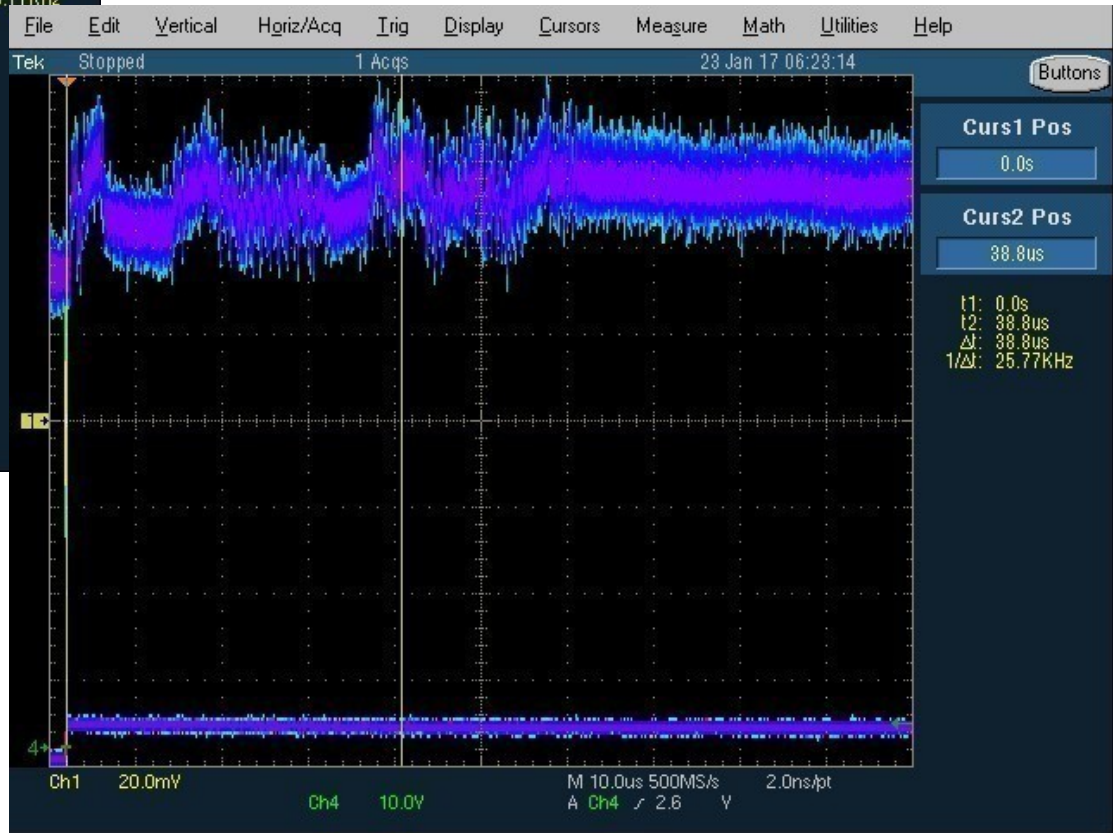
LPC812

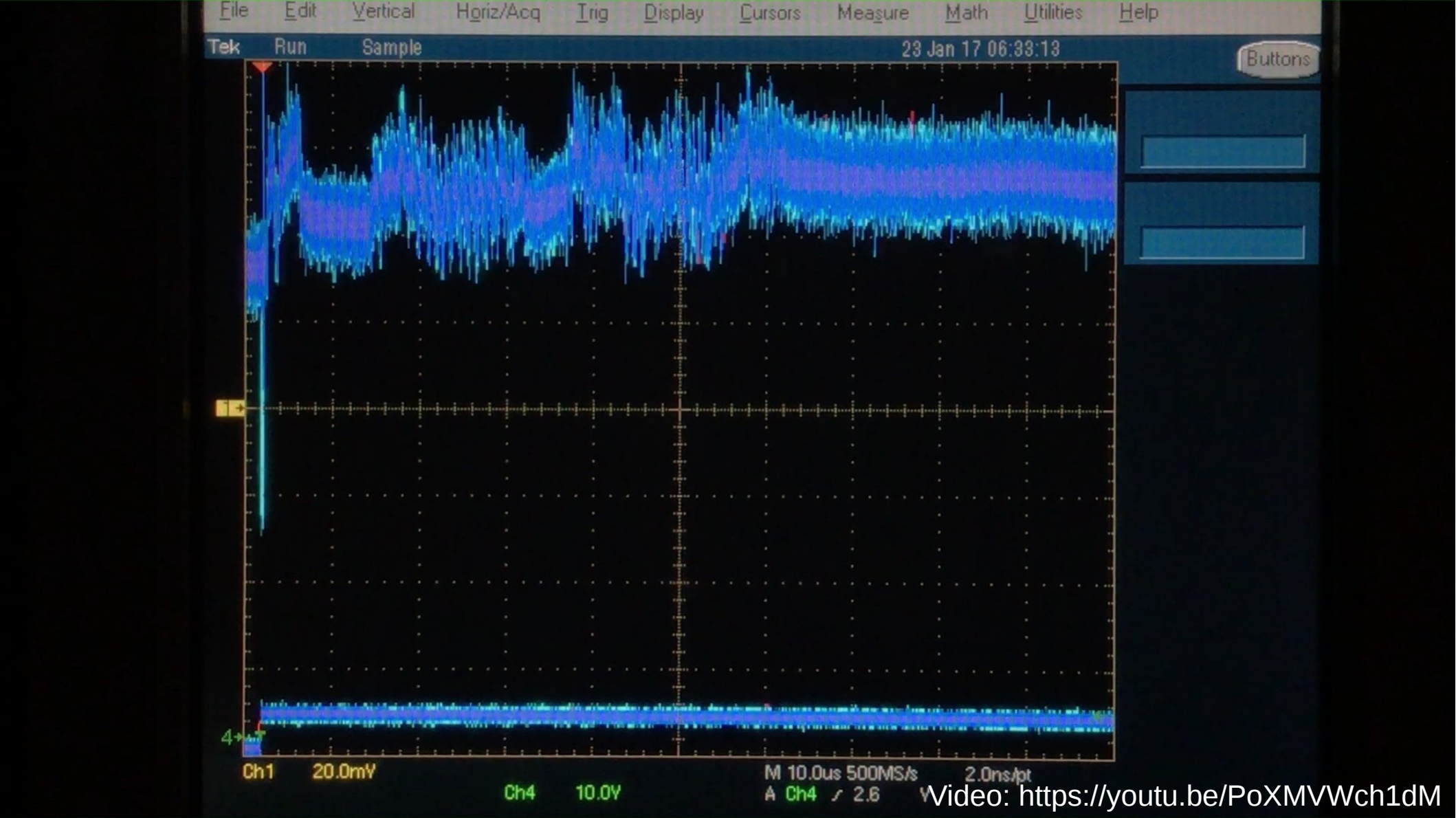
Volage measured across 10 ohm resistor
in series with GND

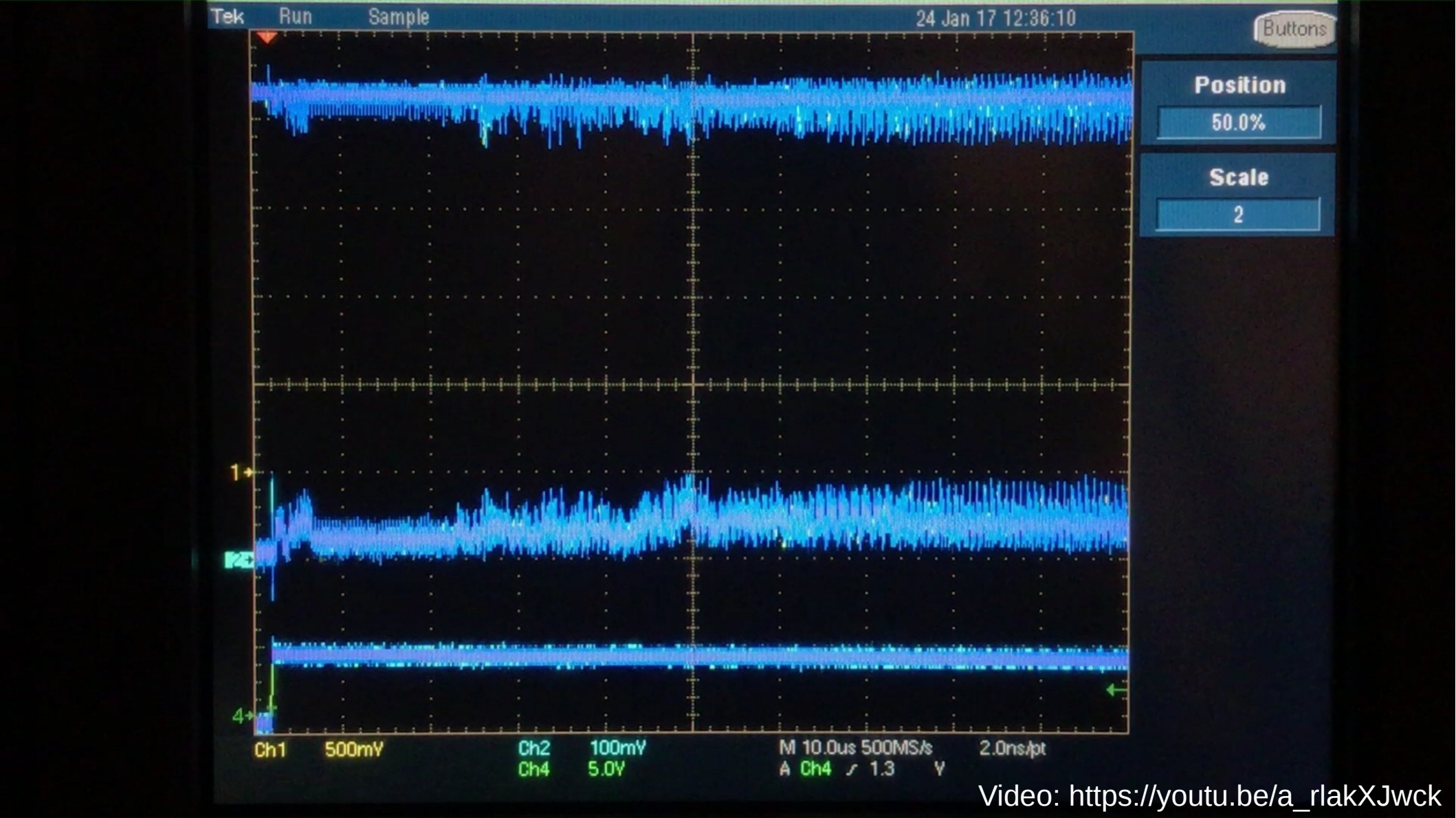


Reset and start application

Reset and start bootloader




File Edit Vertical Horiz/Acq Trig Display Cursors Measure Math Utilities Help
Tek Run Sample
23 Jan 17 06:33:13
Buttons
Ch1 20.0mV
Ch4 10.0V
M 10.0us 500MS/s 2.0ns/pt
A Ch4 ⁄ 2.6
Video: https://youtu.be/PoXMVWch1dM


Tek Run Sample
24 Jan 17 12:36:10
Buttons
Position
50.0%
Scale
2
Ch1 500mV
Ch2 100mV
Ch4 5.0V
M 10.0us 500MS/s
A Ch4 ⟋ 1.3 V
2.0ns/pt
Video: https://youtu.be/a_rlakXJwck


MAXIM
MAX4619CPE
+1232

maxim integrated™

MAX4617/MAX4618/MAX4619

High-Speed, Low-Voltage, CMOS Analog Multiplexers/Switches

General Description

The MAX4617/MAX4618/MAX4619 are high-speed, low-voltage, CMOS analog ICs configured as an 8-channel multiplexer (MAX4617), two 4-channel multiplexers (MAX4618), and three single-pole/double-throw (SPDT) switches (MAX4619).

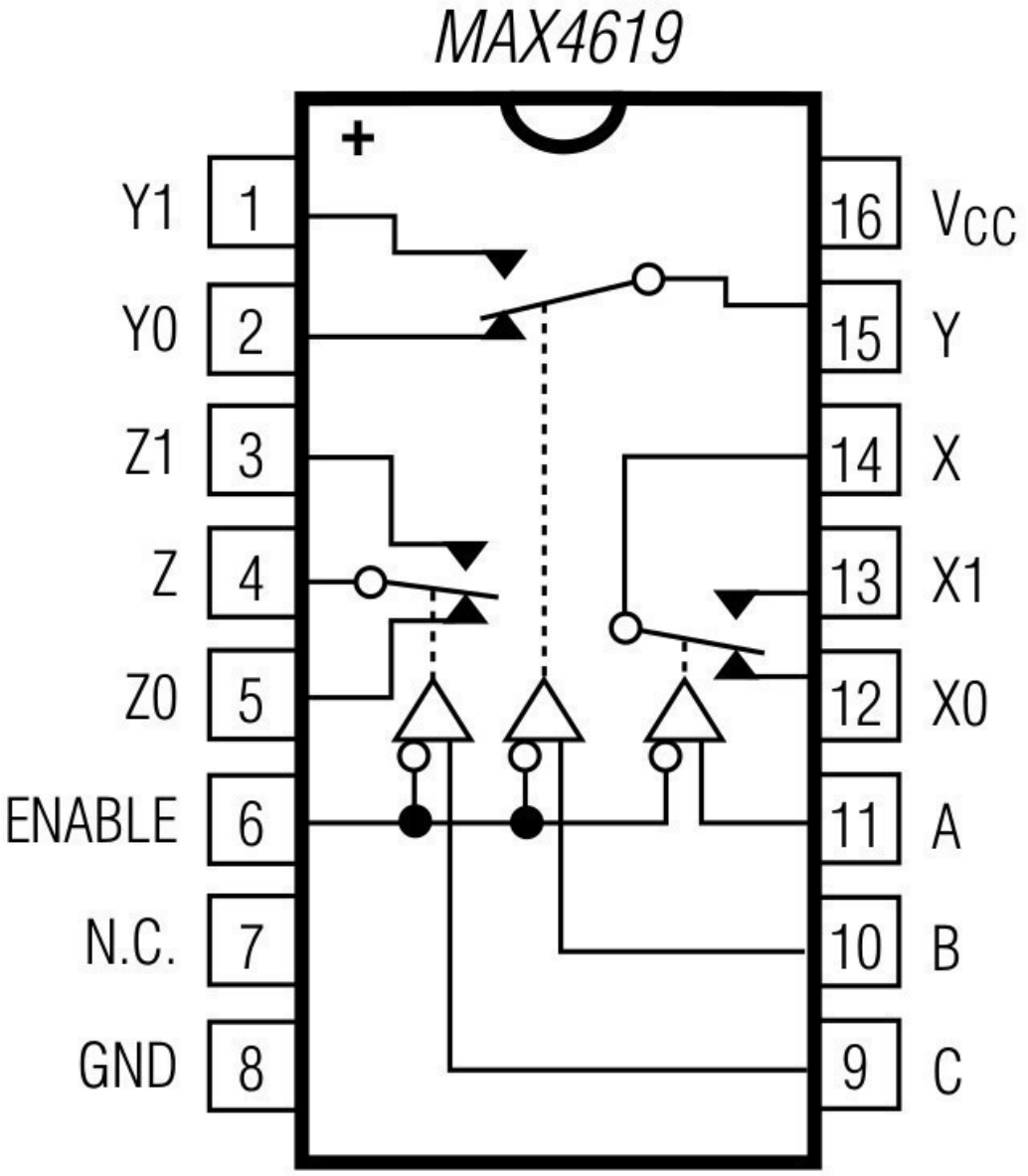
These CMOS devices can operate continuously with a +2V to +5.5V single supply. Each switch can handle rail-to-rail analog signals. The off-leakage current is only 1nA at T_A = +25°C and 10nA at T_A = +85°C.

All digital inputs have 0.8V to 2.4V logic thresholds, ensuring TTL/CMOS-logic compatibility when using a single +5V supply.

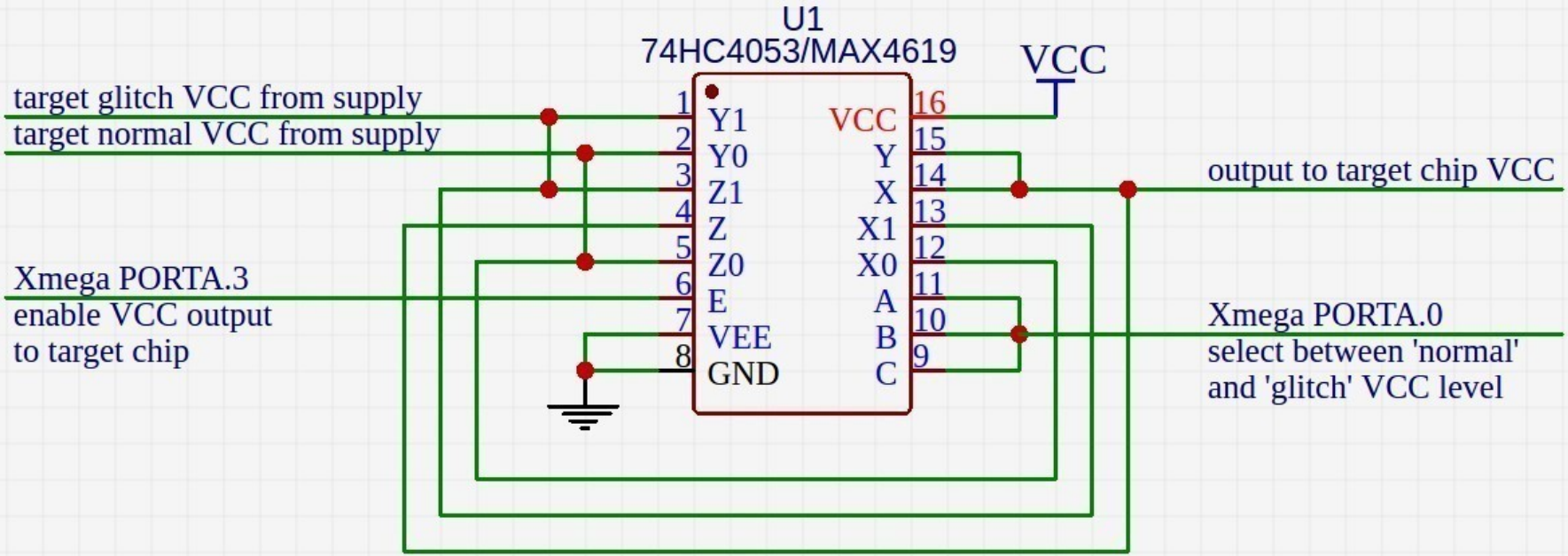
Features

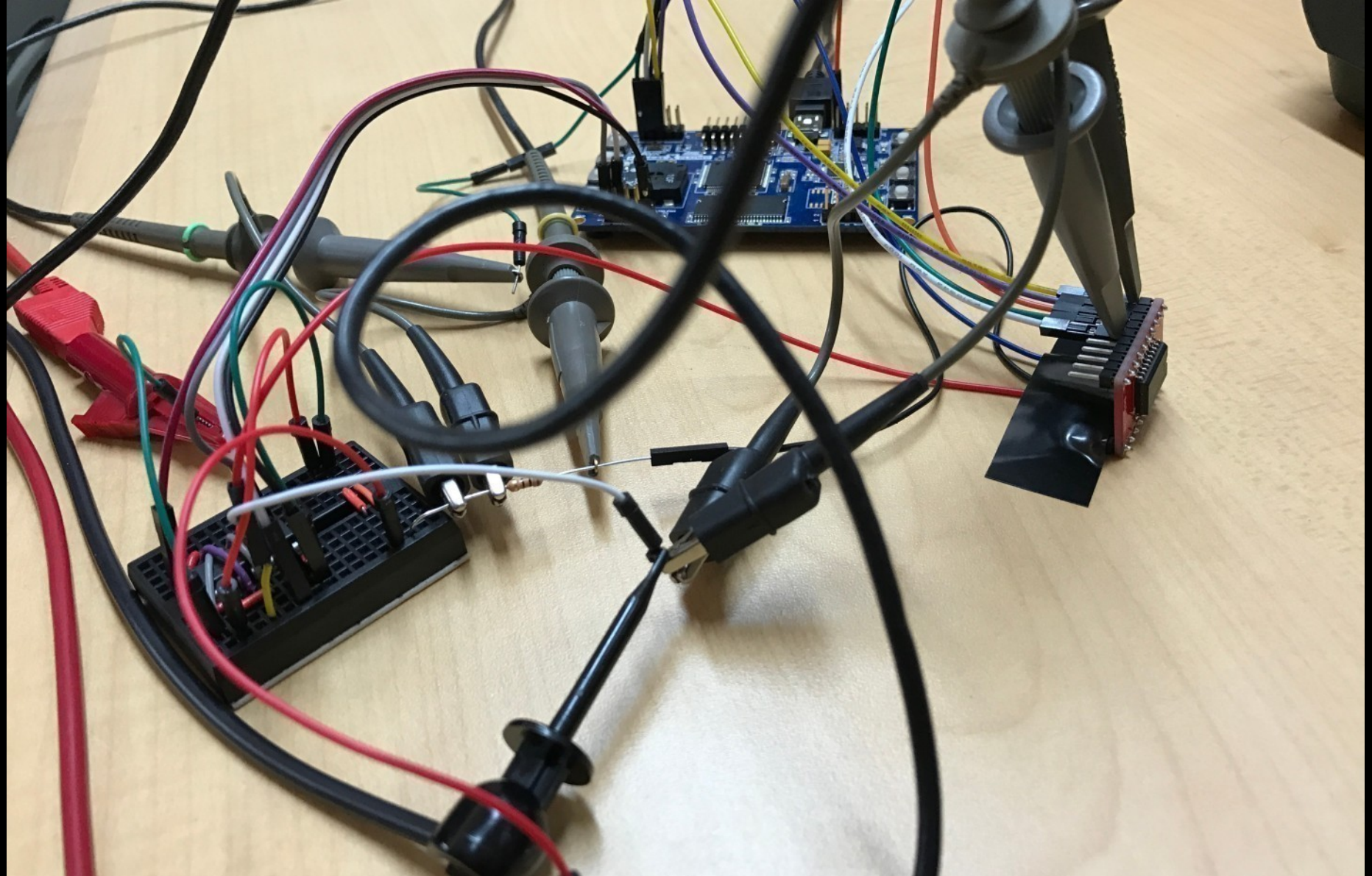
- **Fast Switching Times**
  - **15ns t_{ON}**
  - **10ns t_{OFF}**
- **Pin Compatible with Industry-Standard 74HC4051/74HC4052/74HC4053 and MAX4581/MAX4582/MAX4583**
- **Guaranteed On-Resistance**
  - **10Ω max (+5V Supply)**
  - **20Ω max (+3V Supply)**
- **Guaranteed 1Ω On-Resistance Match Between Channels (single +5V supply)**
- **Guaranteed Low Off-Leakage Current: 1nA at +25°C**
- **Guaranteed Low On-Leakage Current: 1nA at +25°C**
- **+2V to +5.5V Single-Supply Operation**
- **TTL/CMOS-Logic Compatible**

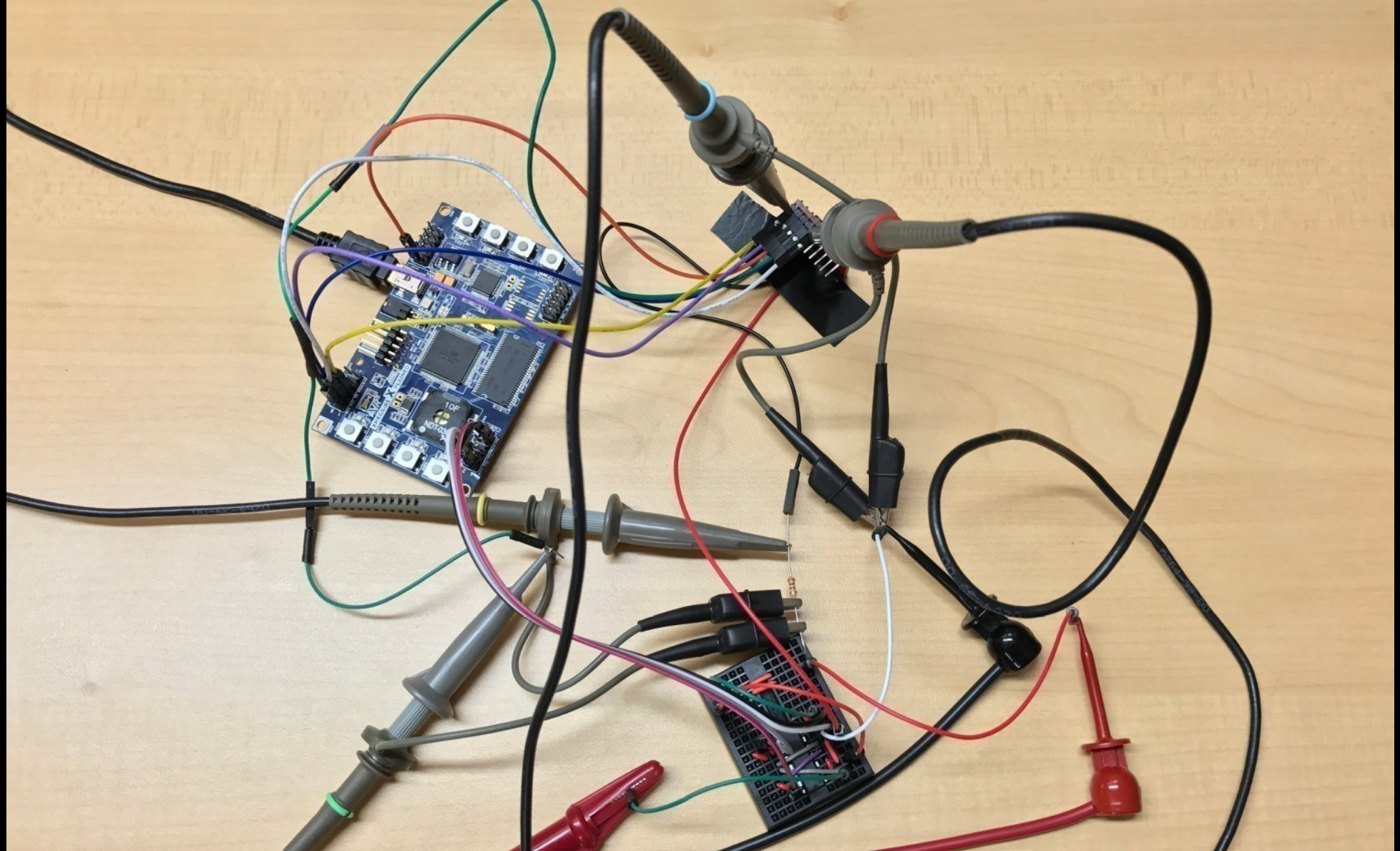
Source: MAX4619 datasheet

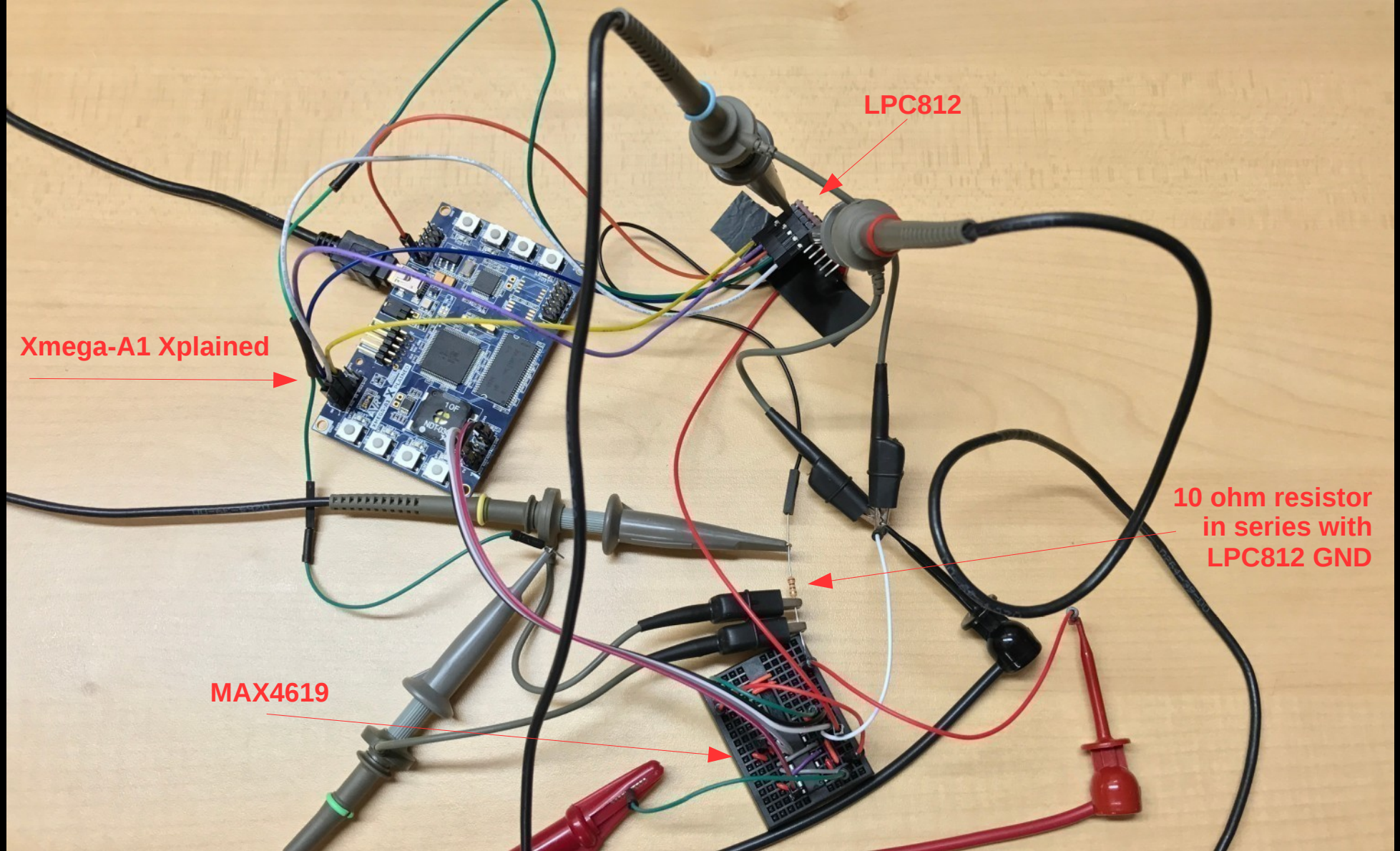


Source: MAX4619 datasheet

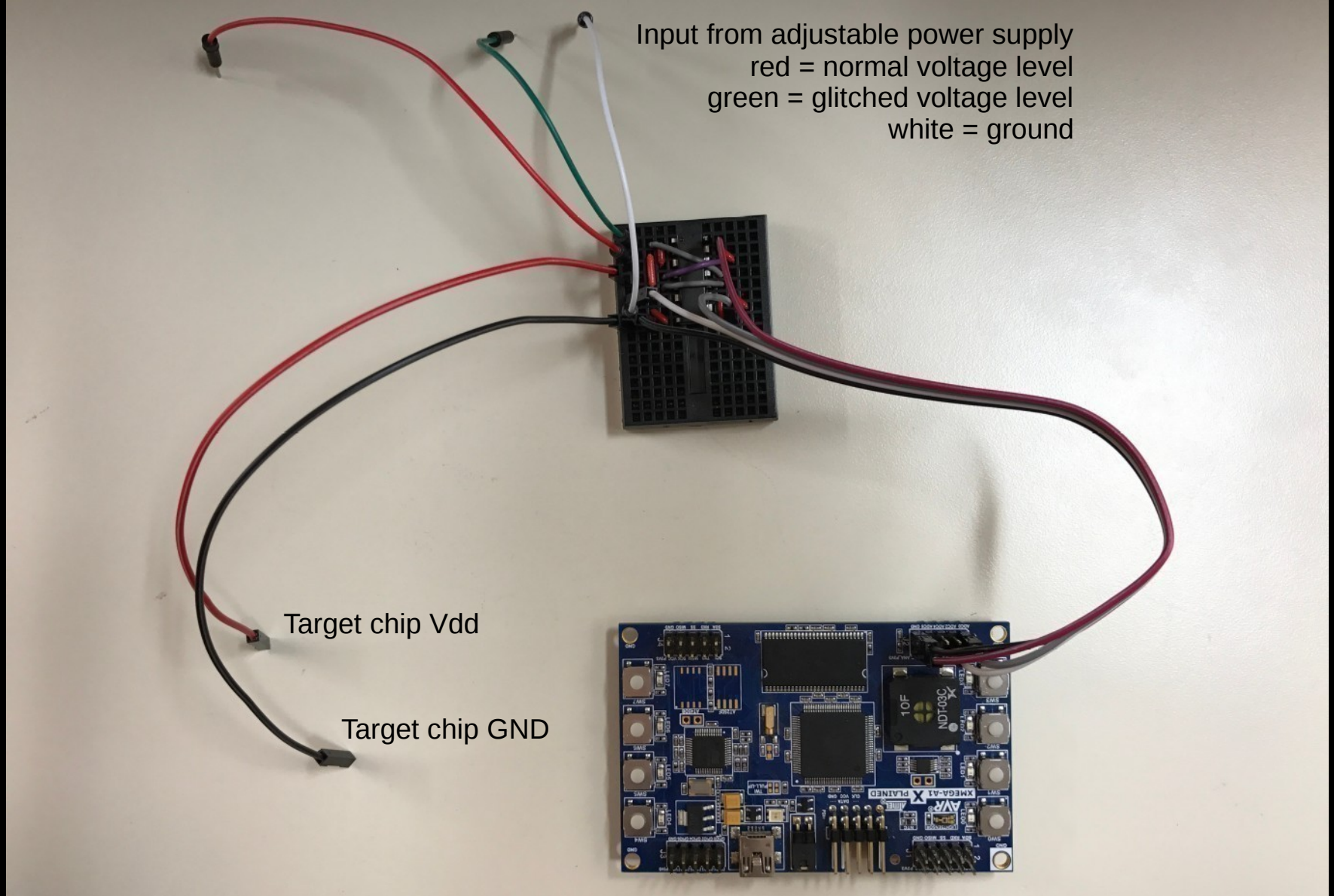

U1
74HC4053/MAX4619
VCC
target glitch VCC from supply
target normal VCC from supply
Xmega PORTA.3
enable VCC output
to target chip
output to target chip VCC
Xmega PORTA.0
select between 'normal'
and 'glitch' VCC level
1 Y1
2 Y0
3 Z1
4 Z
5 Z0
6 E
7 VEE
8 GND
VCC 16
Y 15
X 14
X1 13
X0 12
A 11
B 10
C 9






LPC812
Xmega-A1 Xplained
10 ohm resistor
in series with
LPC812 GND
MAX4619


VIZATEK
MODEL:MPS-3005L-3
A
V
A
V
0.00
01.9
0.00
00.7
LABORATORY DC POWER SUPPLY
CURRENT
VOLTAGE
CURRENT
VOLTAGE
INDEP.
SERIES
PARALLEL
C.C.
C.V.
PAR.
C.C.
C.V.
OVER LOAD 5V/3A
SLAVE
TRACKING
MASTER
POWER
ON
OFF
–
GND
+
–
GND
+
–
+

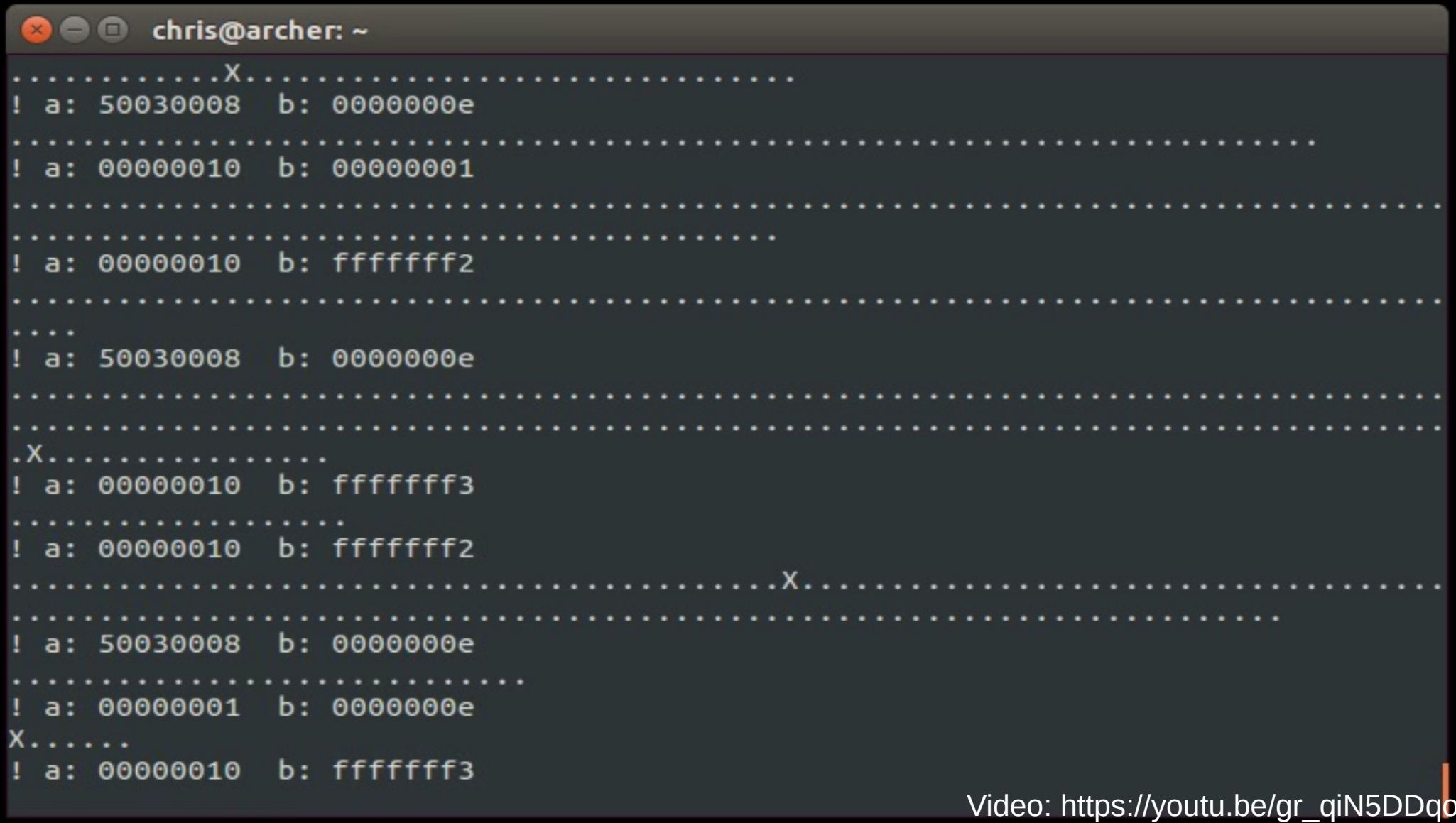

Input from adjustable power supply
red = normal voltage level
green = glitched voltage level
white = ground
Target chip Vdd
Target chip GND

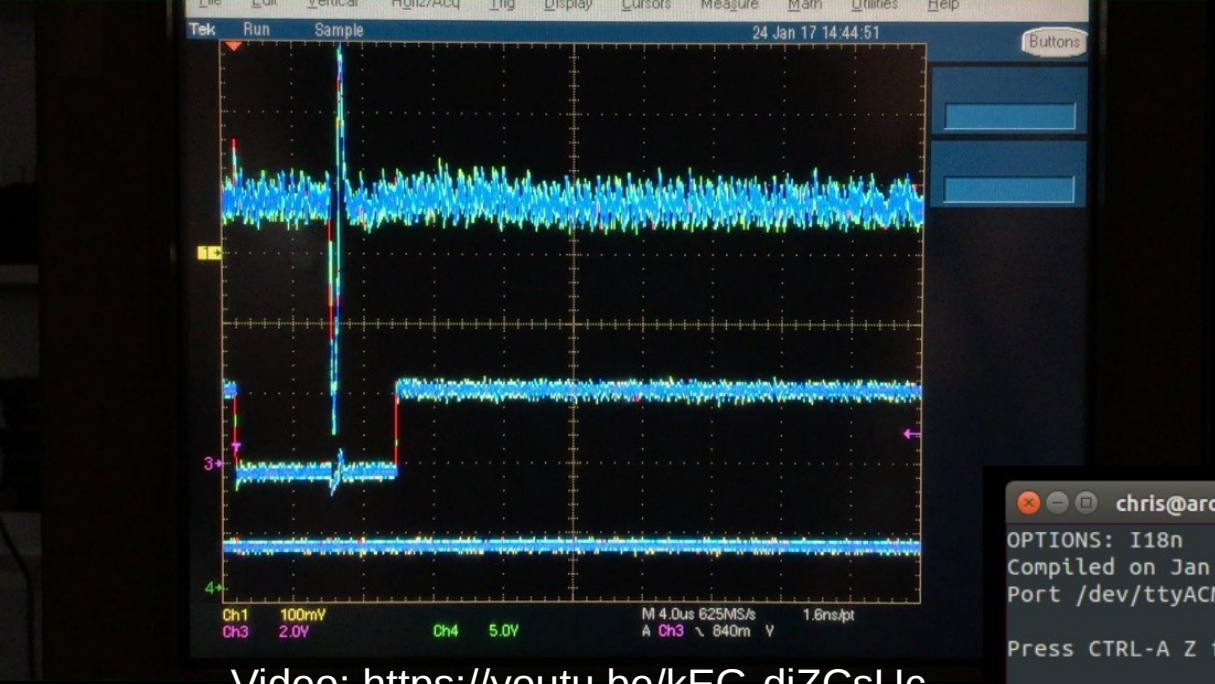
```
//------------------------------------------------------------------
// pin GVCC_EN (port bit 3) must be 0 to enable VCC output to chip
// pin GVCC_SW (port bit 0) is 0 for normal VCC, 1 for glitched VCC
uint8_t glitch_form[8] = {(1<<GVCC_SW), (1<<GVCC_SW), (1<<GVCC_SW), (1<<GVCC_SW), (1<<GVCC_SW), (1<<GVCC_SW), 0, 0};
uint8_t noglitch_form = 0x00;           // GVCC_EN = 0, GVCC_SW = 0
//------------------------------------------------------------------


    uint16_t glitch_delay = 1;          // delay (in 32 MHz cycles) between trigger and glitch


    asm volatile(
        "call DelayCycles" "\n\t"       // call DelayCycles() for single-cycle precision delay
        "out 0x0011, %0" "\n\t"
        "out 0x0011, %1" "\n\t"         // 0x0011 is VPORT0 out
        "out 0x0011, %2" "\n\t"
        "out 0x0011, %3" "\n\t"         // OUT is a single-cycle opcode - executing a series of OUT instructions
        "out 0x0011, %4" "\n\t"         // changes the output port each cycle @ the 32 MHz Xmega clock speed
        "out 0x0011, %5" "\n\t"
        "out 0x0011, %6" "\n\t"         // VCC glitch waveform:
        "out 0x0011, %7" "\n\t"         //
        "out 0x0011, %8" "\n\t"         //    0 0 0 0 0 0 1 1
        :                               // __                 ______
        :                               //   |_____________|
        "r" (glitch_form[0]),           //   ^               ^
        "r" (glitch_form[1]),           //   |---------------|
        "r" (glitch_form[2]),           //   | 8 clk @32 MHz |
        "r" (glitch_form[3]),
        "r" (glitch_form[4]),
        "r" (glitch_form[5]),
        "r" (glitch_form[6]),
        "r" (glitch_form[7]),
        "r" (noglitch_form),
        "z" (glitch_delay)
    );
```

```
// keep repeating glitch-test for() loop
while(1)
{
    b = NUM_GLITCH_LOOPS;                       // initial value of 'b' before glitch loop is NUM_GLITCH_LOOPS

    while( GPIO_PORT0 & (1<<13) )               // wait for PIO0_13 to go low (synchronize with glitcher)
        ;

    while( !(GPIO_PORT0 & (1<<13)) )            // wait for PIO0_13 to go high
        ;

    GPIO_CLRP0 |= (1<<17);                      // pull PIO0_17 low while running for() loop

    for( a=0; a<NUM_GLITCH_LOOPS; a++ )         // NUM_GLITCH_LOOPS is #defined as 0x100
        b--;

    GPIO_SETP0 |= (1<<17);                      // set PIO0_17 high to indicate for() loop has completed

    // expected values:
    // a == NUM_GLITCH_LOOPS (0x100)
    // b == 0
    if( a == NUM_GLITCH_LOOPS && b == 0 )       // no glitch occured - print '.' to indicate 'a' and b' not glitched
        pp_printf(".");
    else                                        // glitch occured - print 'a' and 'b' values
        pp_printf("\n! a: %08X  b: %08X\n", a, b);
}
```

Assembler code generated by GCC

```
 84:main.c         ****
 85:main.c         ****         for( a=0; a<NUM_GLITCH_LOOPS; a++ )
224                       .loc 2 85 0
225 00cc 0023             movs    r3, #0  @ tmp117,                  a = 0
226 00ce 7B60             str r3, [r7, #4]    @ tmp117, a
227 00d0 05E0             b   .L8 @
228                     .L9:
 86:main.c         ****             b--;
229                       .loc 2 86 0 discriminator 3
230 00d2 3B68             ldr r3, [r7]    @ tmp119, b
231 00d4 013B             subs    r3, r3, #1  @ tmp118, tmp119,      decrement 'b'
232 00d6 3B60             str r3, [r7]    @ tmp118, b
 85:main.c         ****             b--;
233                       .loc 2 85 0 discriminator 3
234 00d8 7B68             ldr r3, [r7, #4]    @ tmp121, a
235 00da 0133             adds    r3, r3, #1  @ tmp120, tmp121,      increment 'a'
236 00dc 7B60             str r3, [r7, #4]    @ tmp120, a
237                     .L8:
 85:main.c         ****             b--;
238                       .loc 2 85 0 is_stmt 0 discriminator 1
239 00de 7B68             ldr r3, [r7, #4]    @ tmp122, a            check if a < NUM_GLITCH_LOOPS
240 00e0 0F2B             cmp r3, #15 @ tmp122,
241 00e2 F6DD             ble .L9 @,
```


chris@archer: ~
.....X.....
! a: 50030008 b: 0000000e
! a: 00000010 b: 00000001
! a: 00000010 b: fffffff2
! a: 50030008 b: 0000000e
.X.....
! a: 00000010 b: fffffff3
! a: 00000010 b: fffffff2
.....X.....
! a: 50030008 b: 0000000e
! a: 00000001 b: 0000000e
X.....
! a: 00000010 b: fffffff3
Video: https://youtu.be/gr_qiN5DDqo

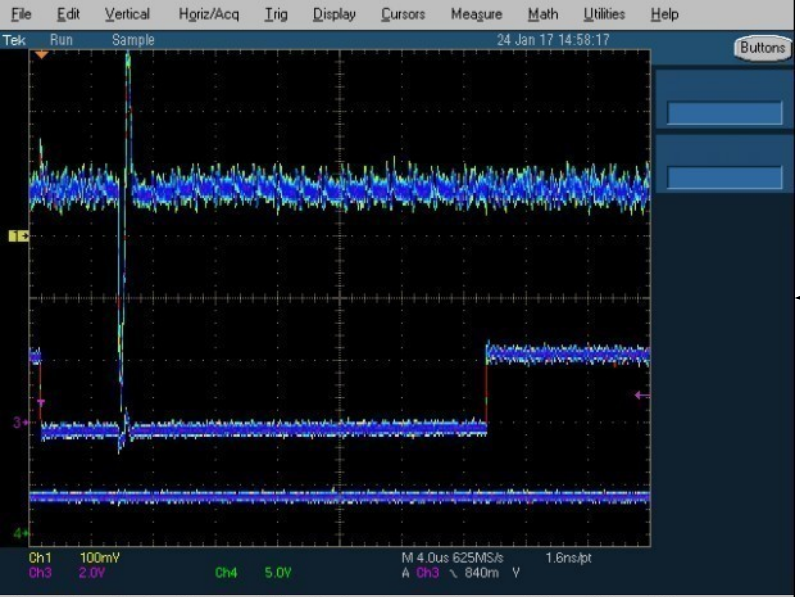


Video: https://youtu.be/kEG-djZCsUc

chris@archer: ~

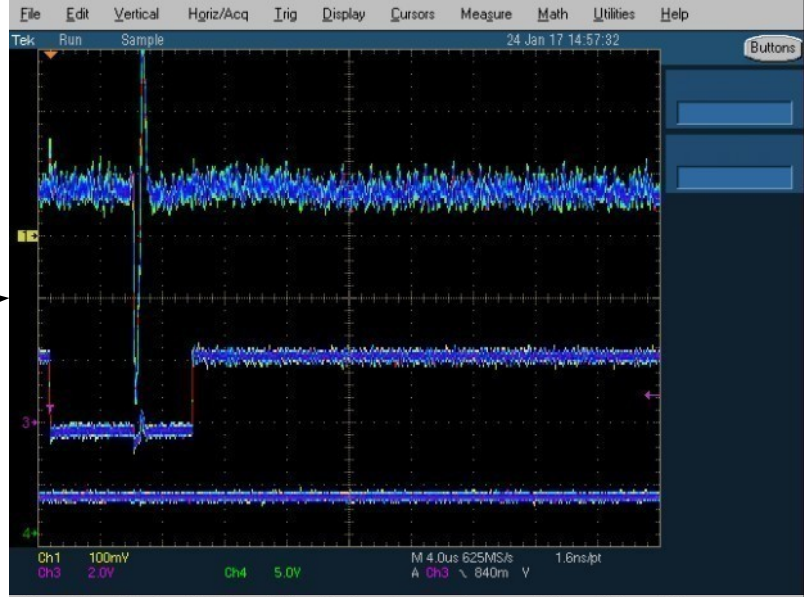
```
OPTIONS: I18n
Compiled on Jan  1 2014, 17:13:19.
Port /dev/ttyACM0, 14:44:46

Press CTRL-A Z for help on special keys


>
>
> target_glitch 70 100 !

VCC glitch waveform:

   0 0 0 0 0 0 1 1
__           ______
  |__________|
  ^          ^
  |----------|
  | 8 clk @32 MHz |

Using delay between trigger and glitch: 70
................................
! a: 50030008  b: 0000000e
>
```

Video: https://youtu.be/r8-1TAQS49c

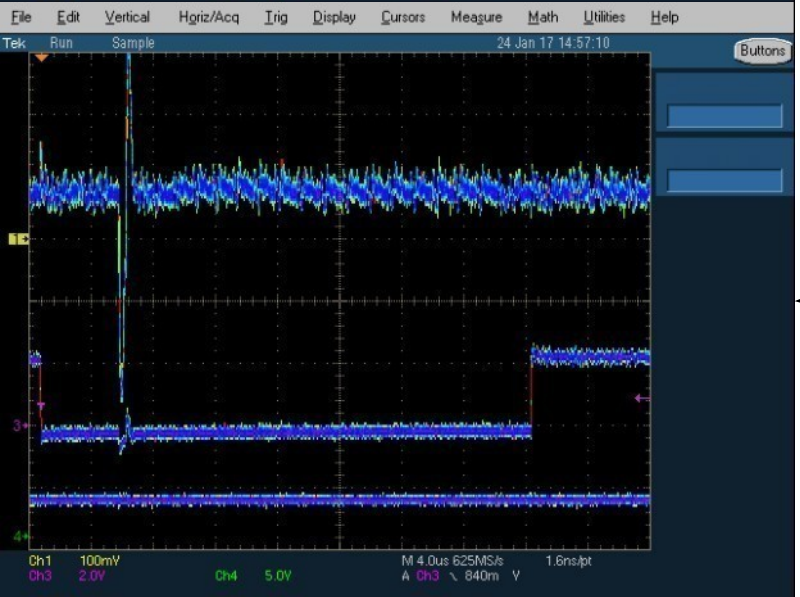


Loop executes normally (no glitch)

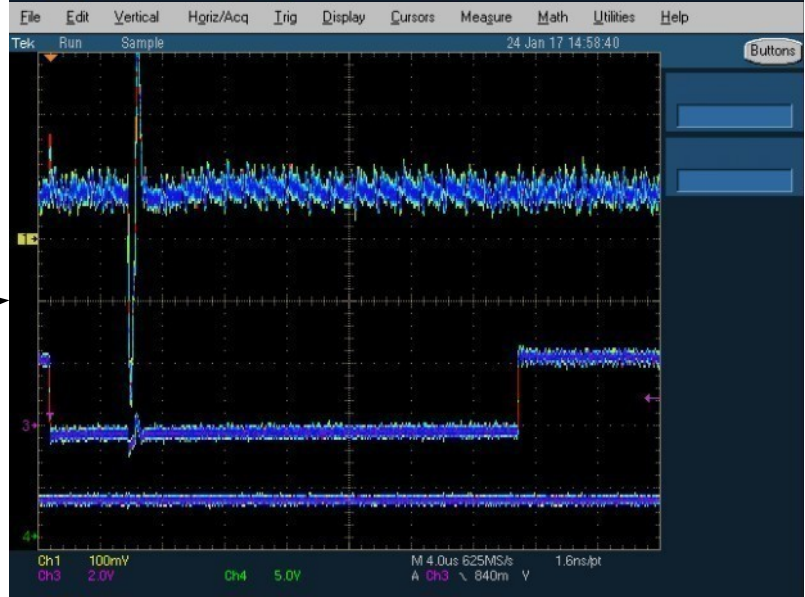
```
a: 00000010
b: 00000000
```



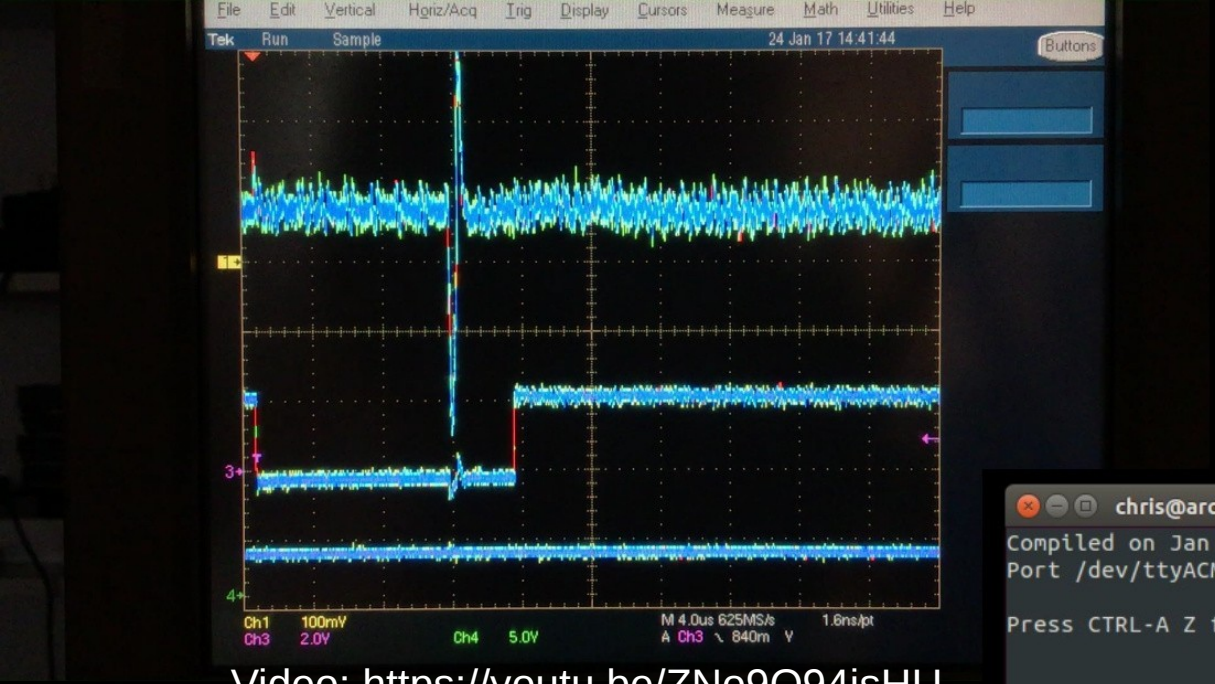

```
a: 50030008
b: 0000000e
```




```
a: 00000010
b: fffffff1
```




```
a: 00000010
b: fffffff2
```

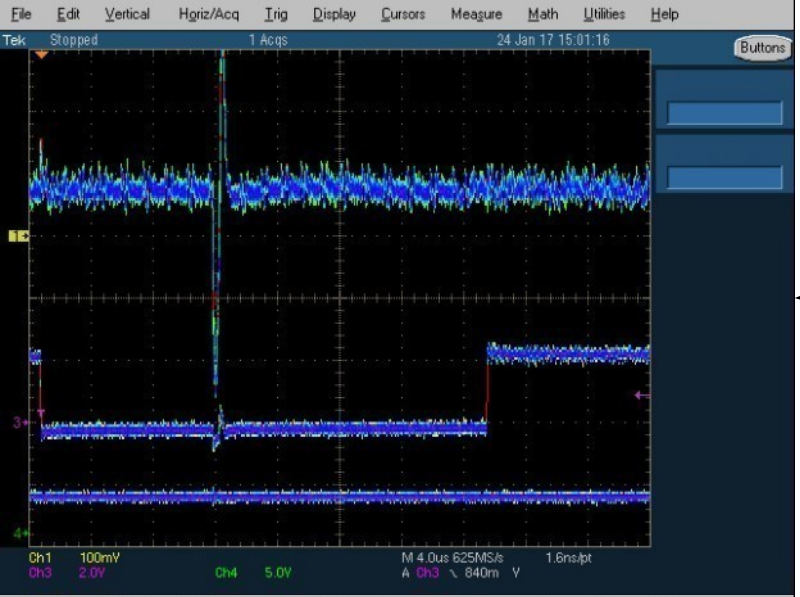


Video: https://youtu.be/ZNo9O94isHU

chris@archer: ~

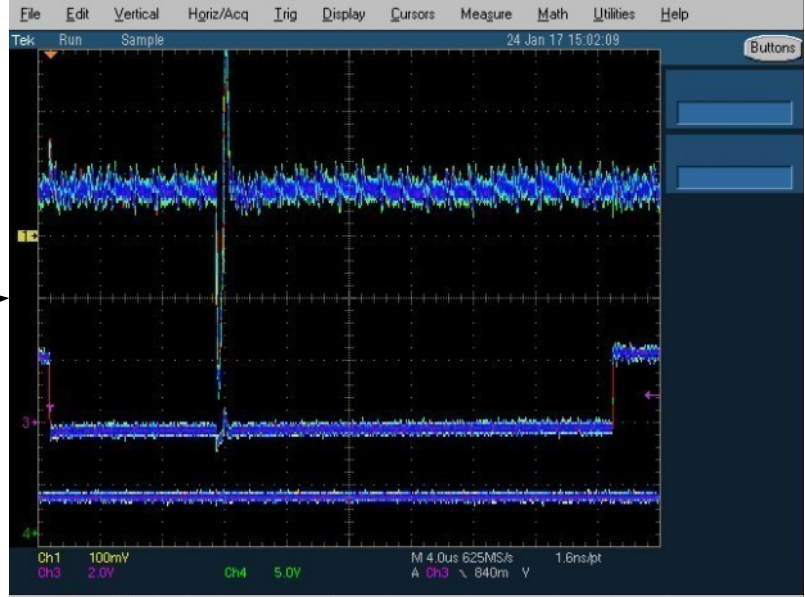
```
Compiled on Jan  1 2014, 17:13:19.
Port /dev/ttyACM0, 14:41:30

Press CTRL-A Z for help on special keys


>
>
>
> target_glitch 260 100 !

VCC glitch waveform:

   0 0 0 0 0 0 1 1
__           ______
  |___________|
  ^           ^
  |-----------|
  | 8 clk @32 MHz |

Using delay between trigger and glitch: 260
.....................................................
! a: 50030008  b: 0000000a
>
```

Video: https://youtu.be/IJWCpp7jgcs

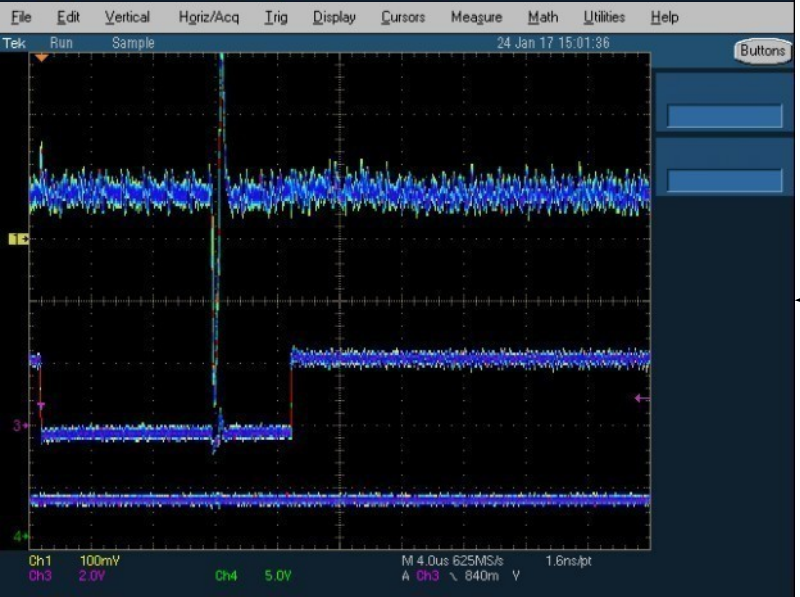


Loop executes normally (no glitch)

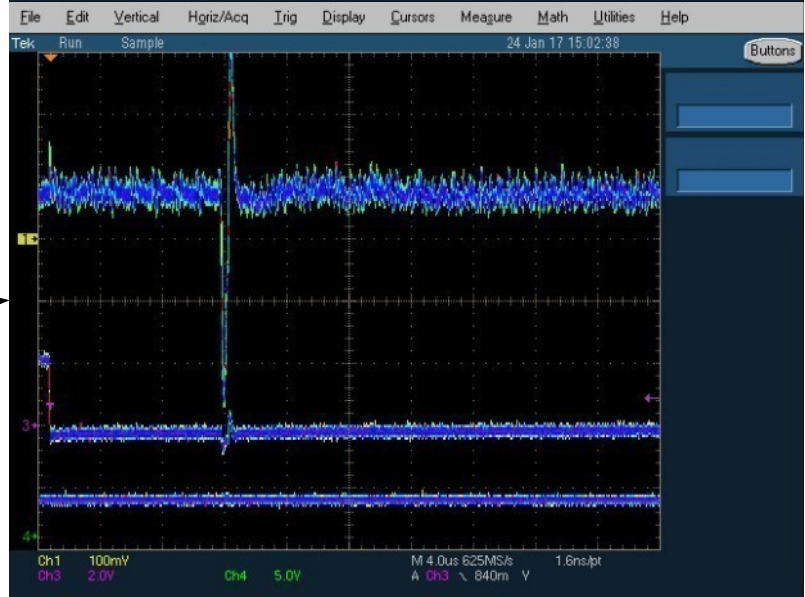
```
a: 00000010
b: 00000000
```



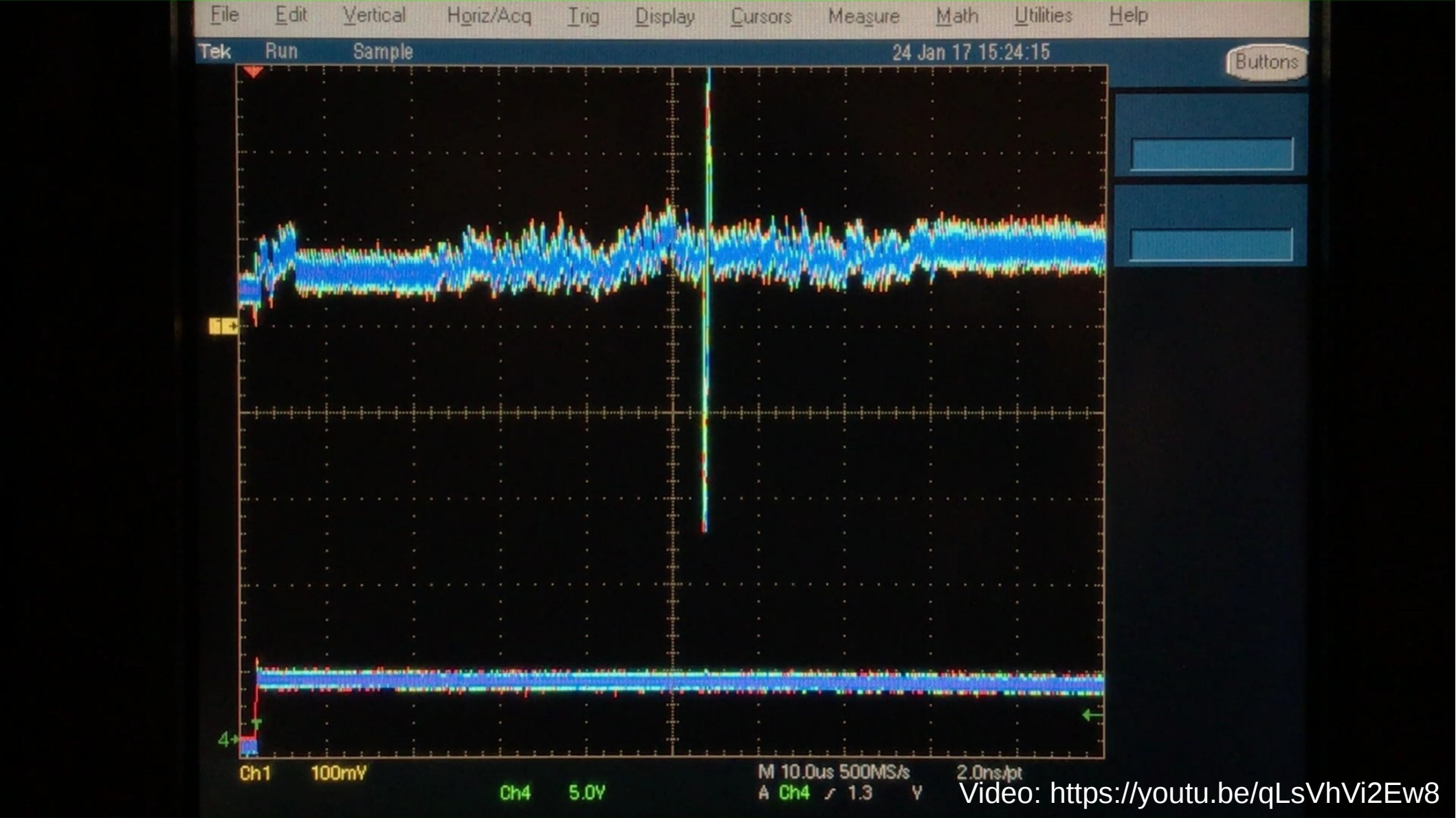

```
a: 00000010
b: fffffffc
```

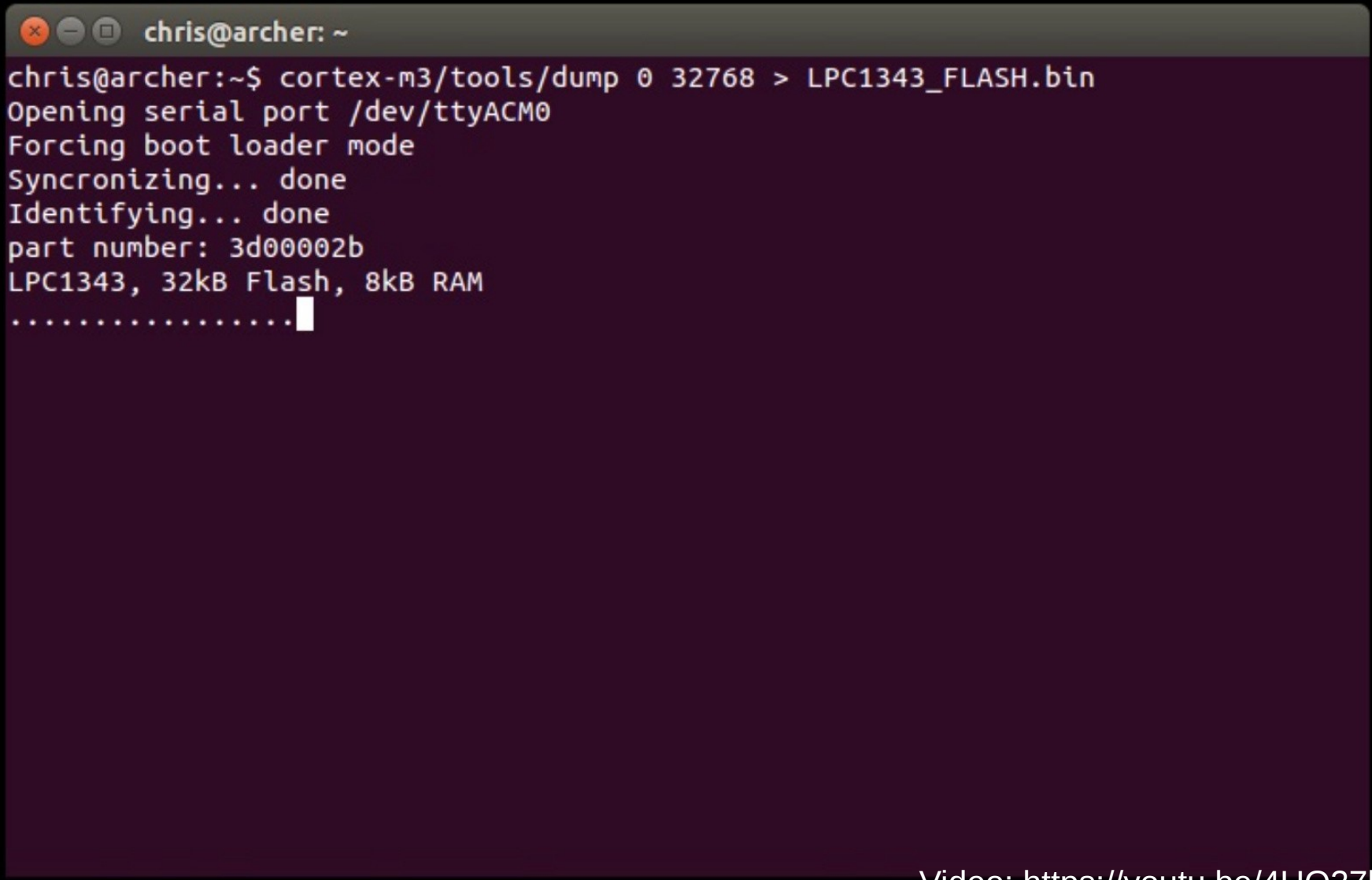



```
a: 50030009
b: 00000009
```



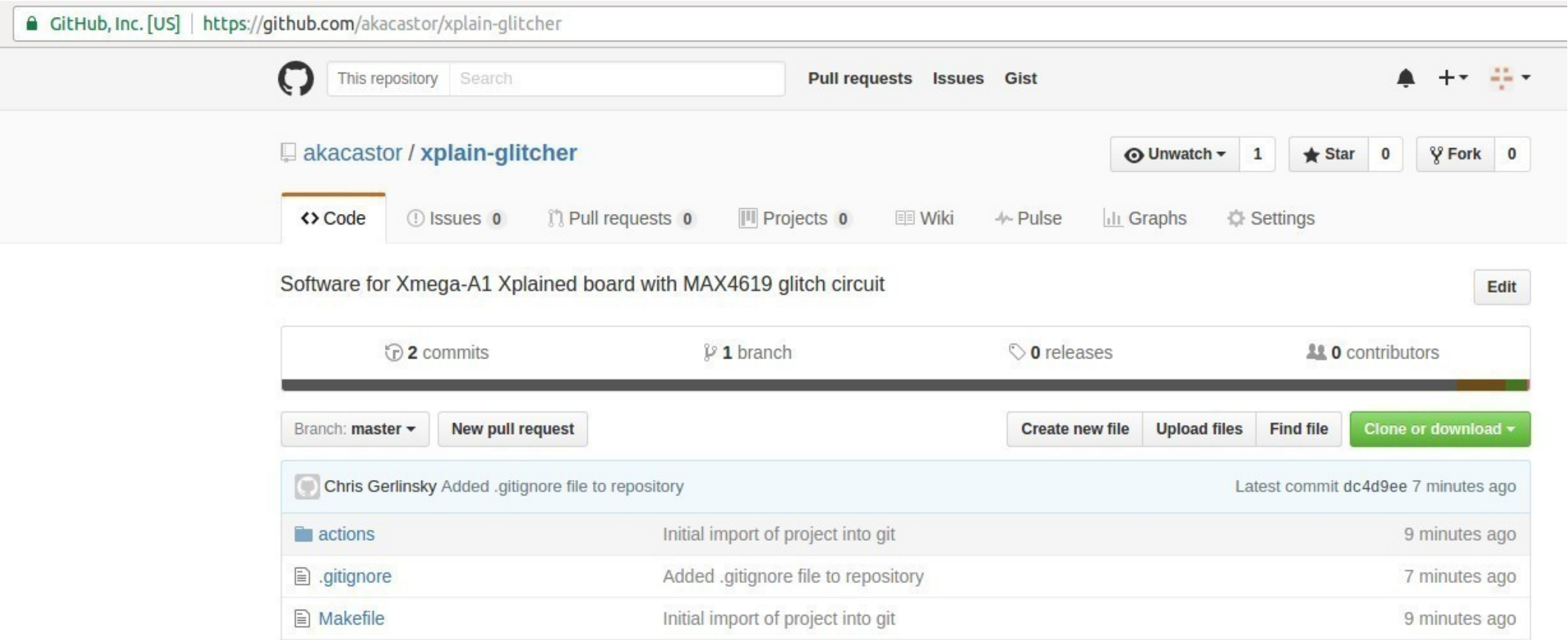

```
a: 00000021
b: 100003fc
```


File Edit Vertical Horiz/Acq Trig Display Cursors Measure Math Utilities Help
Tek Run Sample
24 Jan 17 15:24:15
Buttons
Ch1 100mV
Ch4 5.0V
M 10.0us 500MS/s
A Ch4 ⁄ 1.3 V
2.0ns/pt
Video: https://youtu.be/qLsVhVi2Ew8



Video: https://youtu.be/4UQ27hMuwNw

https://github.com/akacastor/xplain-glitcher



```
void translate_rd( uint32_t offset, uint32_t length)
{
  uint32_t i;
  uint8_t data;
  uint8_t * firmware;
  firmware = (uint8_t *)USER_FLASH_START;
  for ( i = 0; i<length; i++)
  {
    if (offset < BOOT_SECT_SIZE)
    {

      switch (offset)
      {
        case 19:
          data = (uint8_t)(MSC_BlockCount & 0xFF);
        break;
        case 20:
          data = (uint8_t)((MSC_BlockCount >> 8) & 0xFF);
        break;
        case 510:
          data = 0x55;
        break;
        case 511:
          data = 0xAA;
        break;
        default:
        if ( offset > 29 )
        {
          data = 0x0;
        }
        else
        {
          data = BootSect[offset];
        }
        break;
      }

    }
    else if (offset < (BOOT_SECT_SIZE + FAT_SIZE + ROOT_DIR_SIZE))
    {
      data = Fat_RootDir[offset - BOOT_SECT_SIZE];
    }
    else
    {
      if( crp == NOCRP )
      {
        data = *(firmware + (offset - (BOOT_SECT_SIZE + FAT_SIZE + ROOT_DIR_SIZE)));
      }
      else
      {
        data = 0x0;
      }
    }
    Memory[i] = data;
    offset++;
  }
}
```

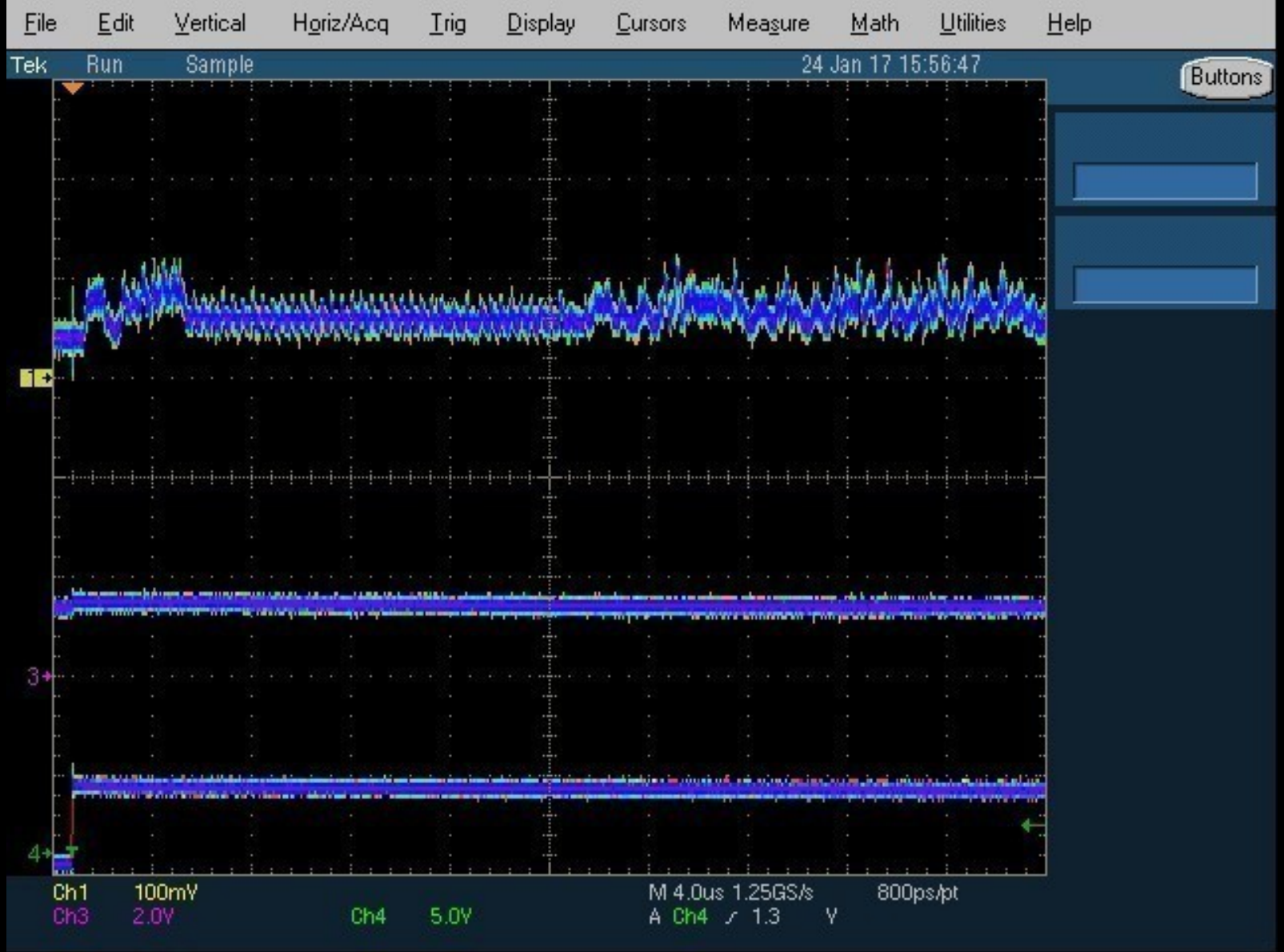
sbl_config.h:

```
#define CRP1  0x12345678
#define CRP2  0x87654321
#define CRP3  0x43218765
#define NOCRP 0x11223344
```

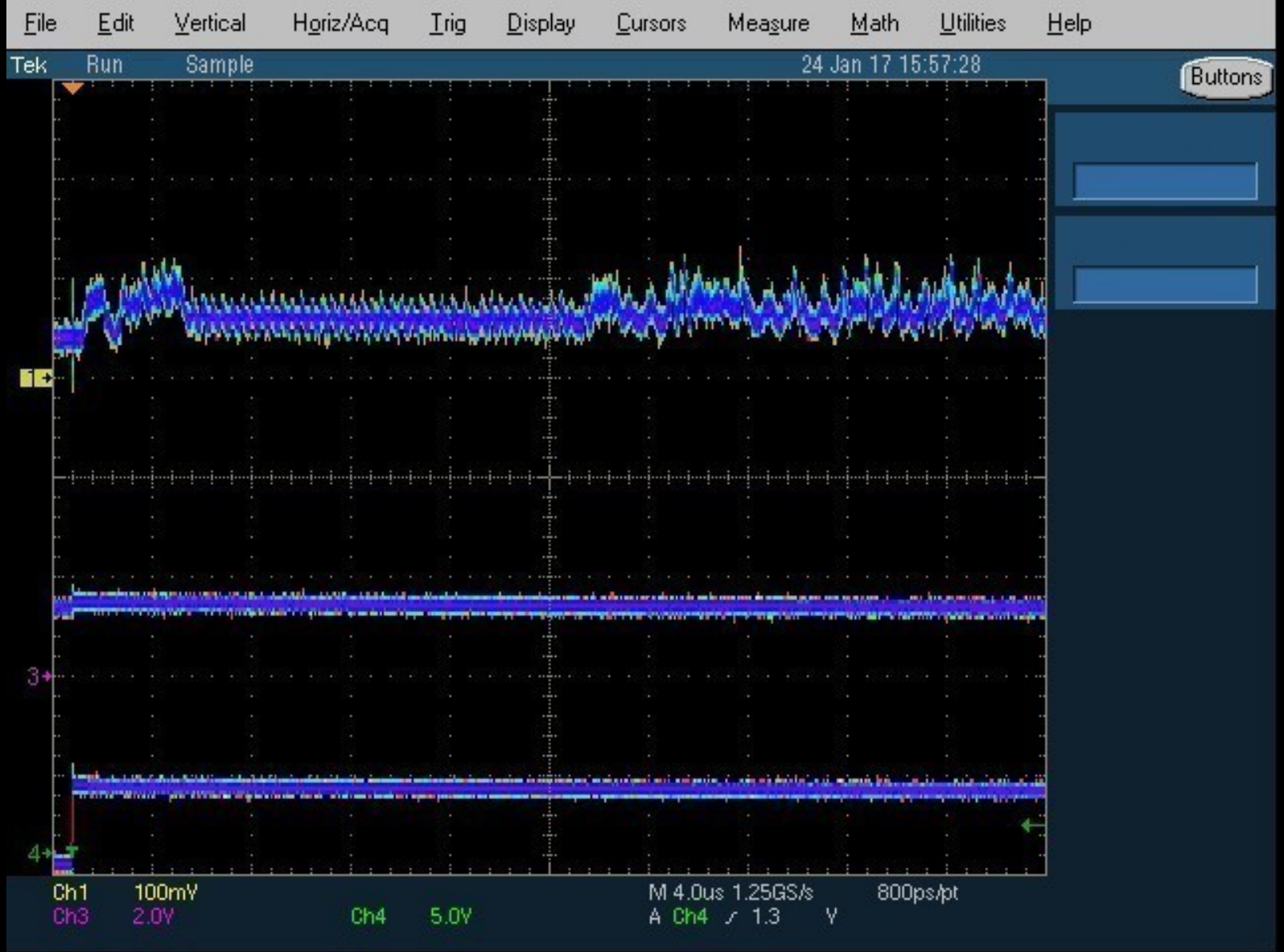
Readout only allowed if CRP word == NOCRP

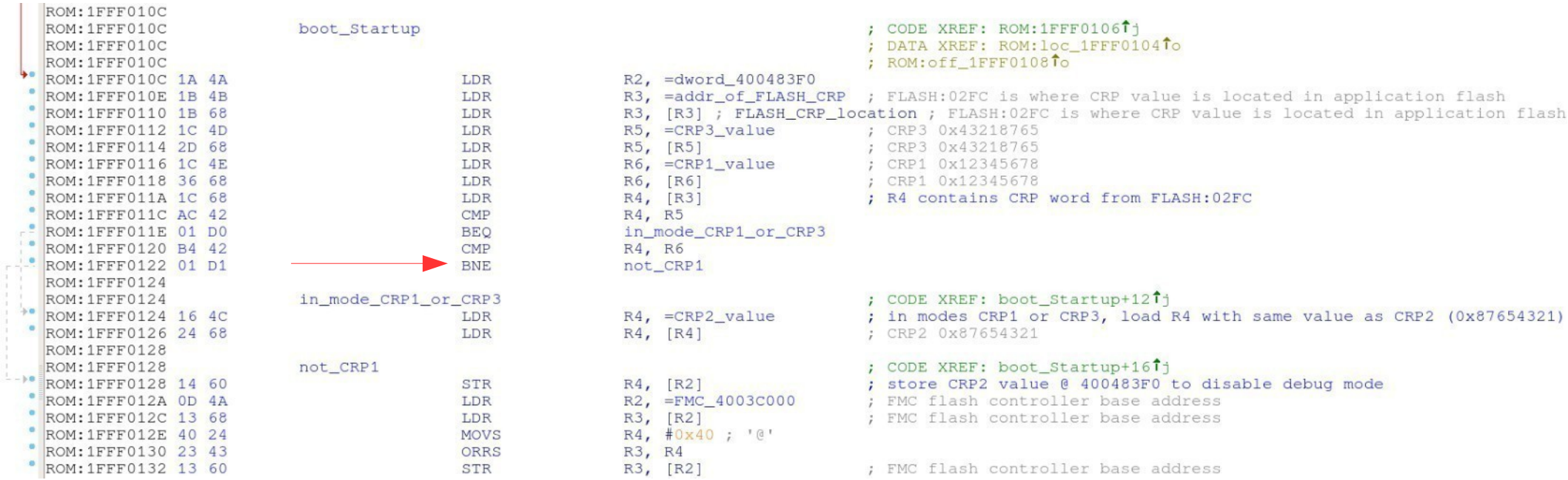
Source: AN10866 LPC1700 Secondary USB bootloader

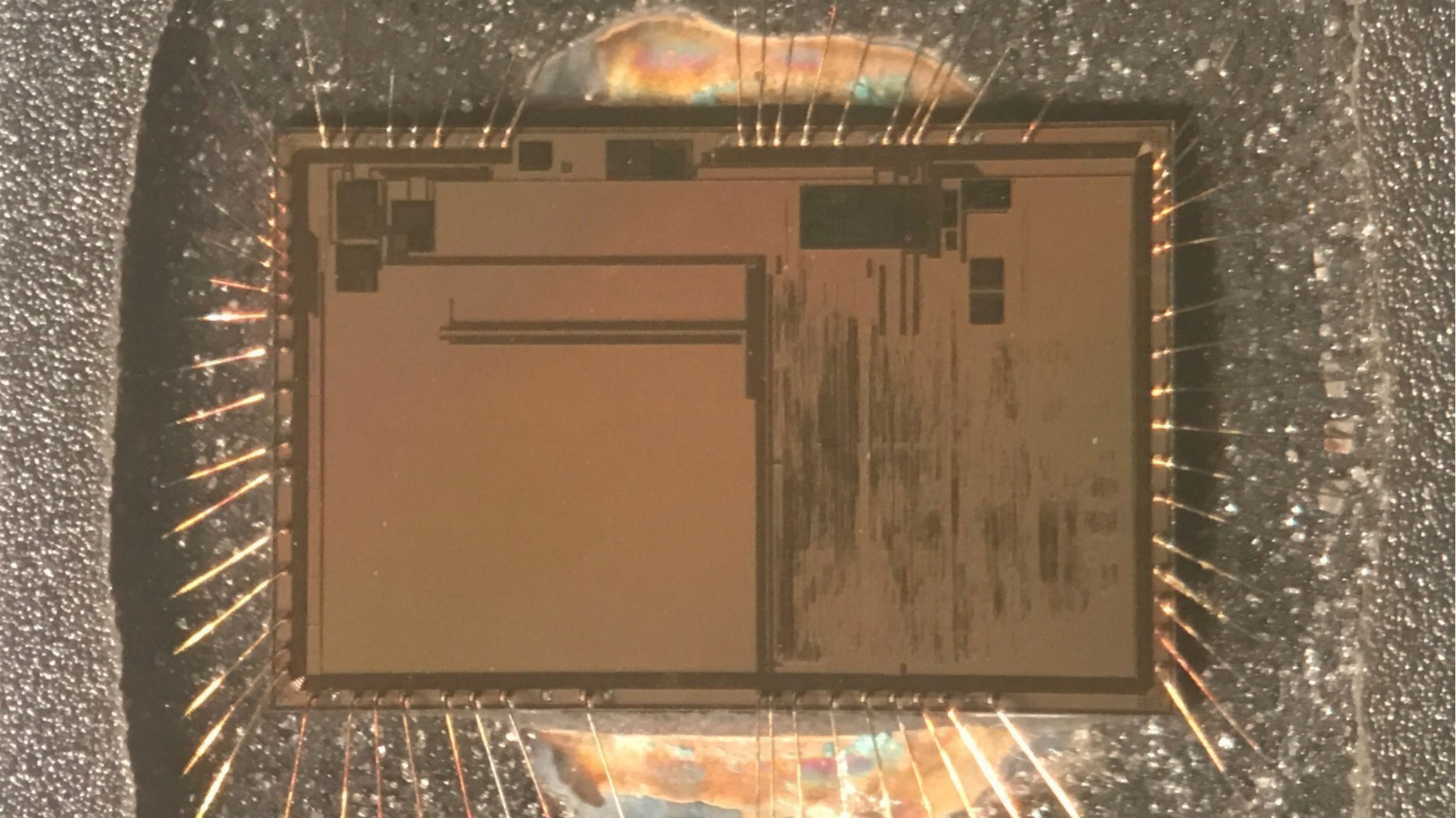
No CRP


File Edit Vertical Horiz/Acq Trig Display Cursors Measure Math Utilities Help
Tek Run Sample
24 Jan 17 15:56:47
Buttons
Ch1 100mV
Ch3 2.0V
Ch4 5.0V
M 4.0us 1.25GS/s
A Ch4 ⁄ 1.3 V
800ps/pt

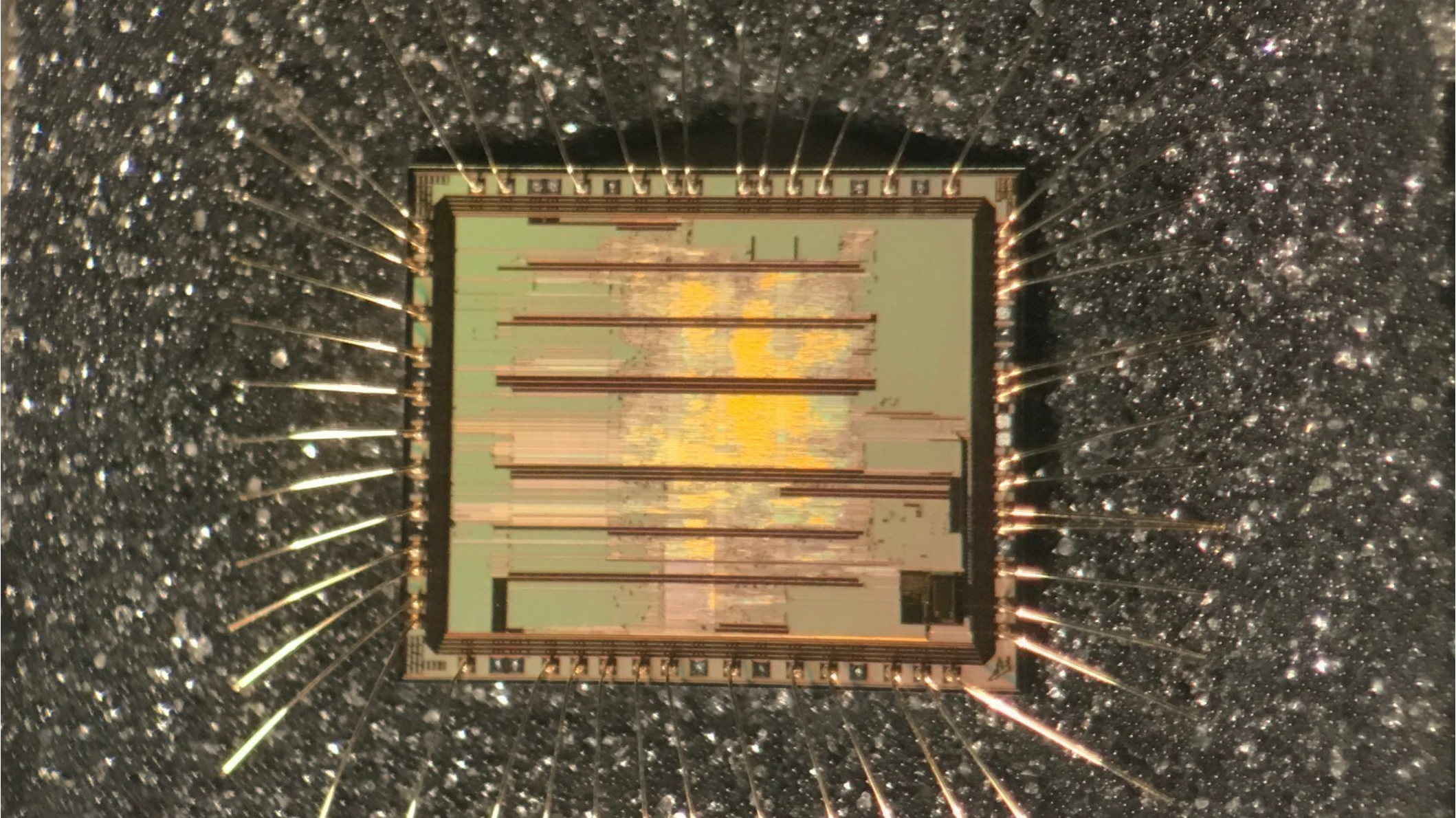
CRP1

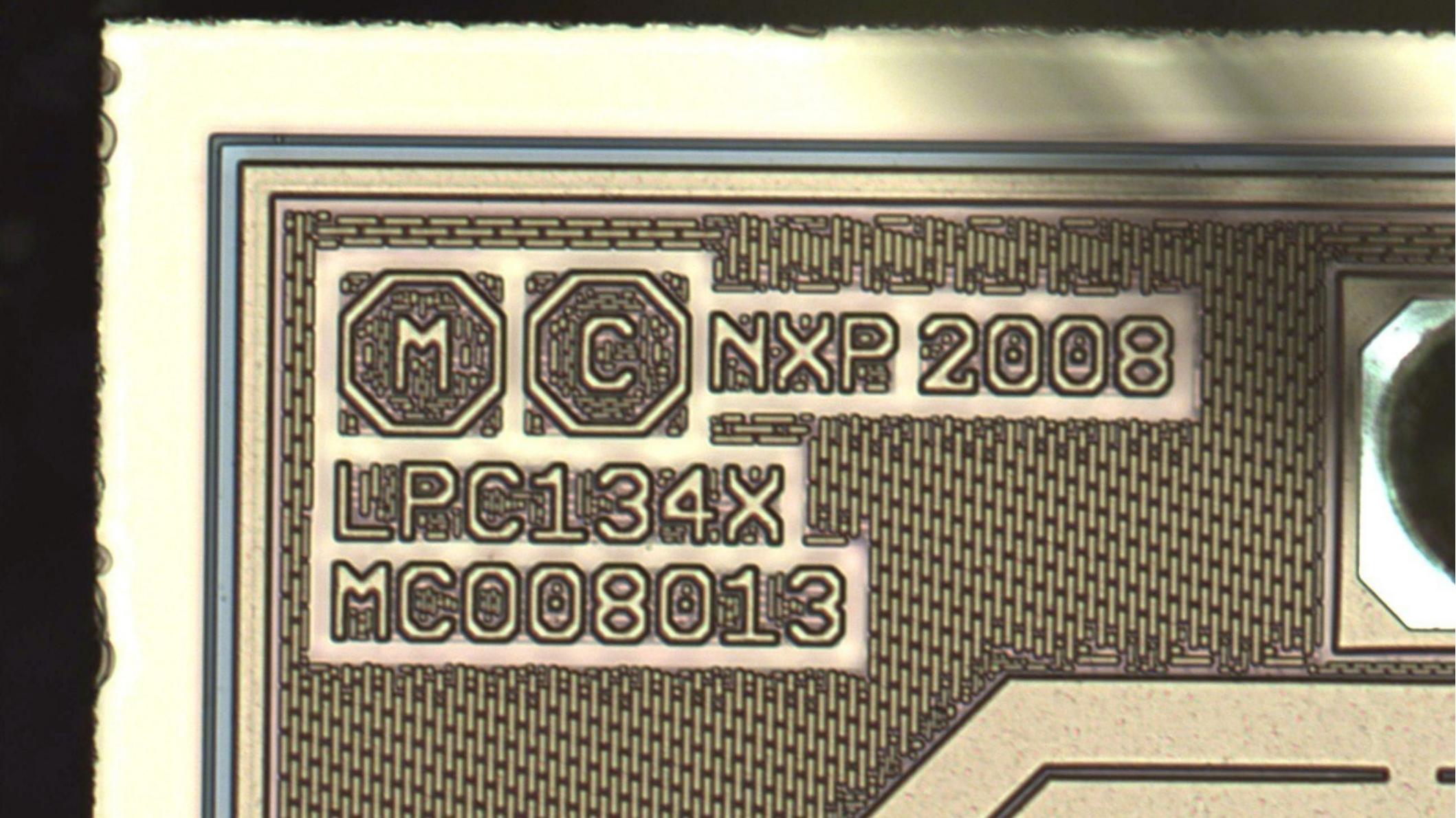

File Edit Vertical Horiz/Acq Trig Display Cursors Measure Math Utilities Help
Tek Run Sample
24 Jan 17 15:57:28
Buttons
Ch1 100mV
Ch3 2.0V
Ch4 5.0V
M 4.0us 1.25GS/s
800ps/pt
A Ch4 ⁄ 1.3 V


ROM:1FFF010C
ROM:1FFF010C boot_Startup ; CODE XREF: ROM:1FFF0106↑j
ROM:1FFF010C ; DATA XREF: ROM:loc_1FFF0104↑o
ROM:1FFF010C ; ROM:off_1FFF0108↑o
ROM:1FFF010C 1A 4A LDR R2, =dword_400483F0
ROM:1FFF010E 1B 4B LDR R3, =addr_of_FLASH_CRP ; FLASH:02FC is where CRP value is located in application flash
ROM:1FFF0110 1B 68 LDR R3, [R3] ; FLASH_CRP_location ; FLASH:02FC is where CRP value is located in application flash
ROM:1FFF0112 1C 4D LDR R5, =CRP3_value ; CRP3 0x43218765
ROM:1FFF0114 2D 68 LDR R5, [R5] ; CRP3 0x43218765
ROM:1FFF0116 1C 4E LDR R6, =CRP1_value ; CRP1 0x12345678
ROM:1FFF0118 36 68 LDR R6, [R6] ; CRP1 0x12345678
ROM:1FFF011A 1C 68 LDR R4, [R3] ; R4 contains CRP word from FLASH:02FC
ROM:1FFF011C AC 42 CMP R4, R5
ROM:1FFF011E 01 D0 BEQ in_mode_CRP1_or_CRP3
ROM:1FFF0120 B4 42 CMP R4, R6
ROM:1FFF0122 01 D1 BNE not_CRP1
ROM:1FFF0124
ROM:1FFF0124 in_mode_CRP1_or_CRP3 ; CODE XREF: boot_Startup+12↑j
ROM:1FFF0124 16 4C LDR R4, =CRP2_value ; in modes CRP1 or CRP3, load R4 with same value as CRP2 (0x87654321)
ROM:1FFF0126 24 68 LDR R4, [R4] ; CRP2 0x87654321
ROM:1FFF0128
ROM:1FFF0128 not_CRP1 ; CODE XREF: boot_Startup+16↑j
ROM:1FFF0128 14 60 STR R4, [R2] ; store CRP2 value @ 400483F0 to disable debug mode
ROM:1FFF012A 0D 4A LDR R2, =FMC_4003C000 ; FMC flash controller base address
ROM:1FFF012C 13 68 LDR R3, [R2] ; FMC flash controller base address
ROM:1FFF012E 40 24 MOVS R4, #0x40 ; '@'
ROM:1FFF0130 23 43 ORRS R3, R4
ROM:1FFF0132 13 60 STR R3, [R2] ; FMC flash controller base address




MC005018
LPC2148




NXP 2008
LPC134X
MC008013