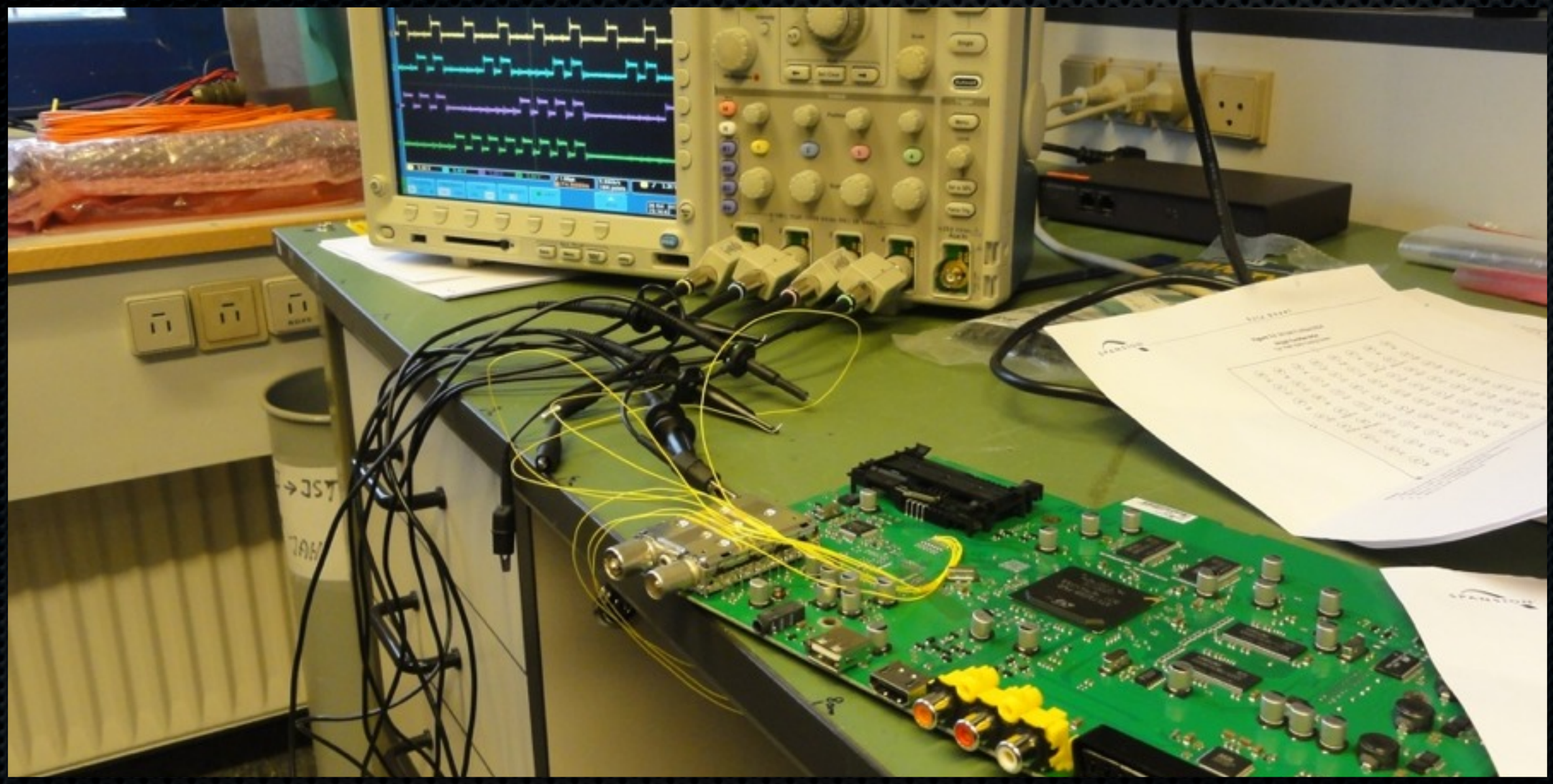
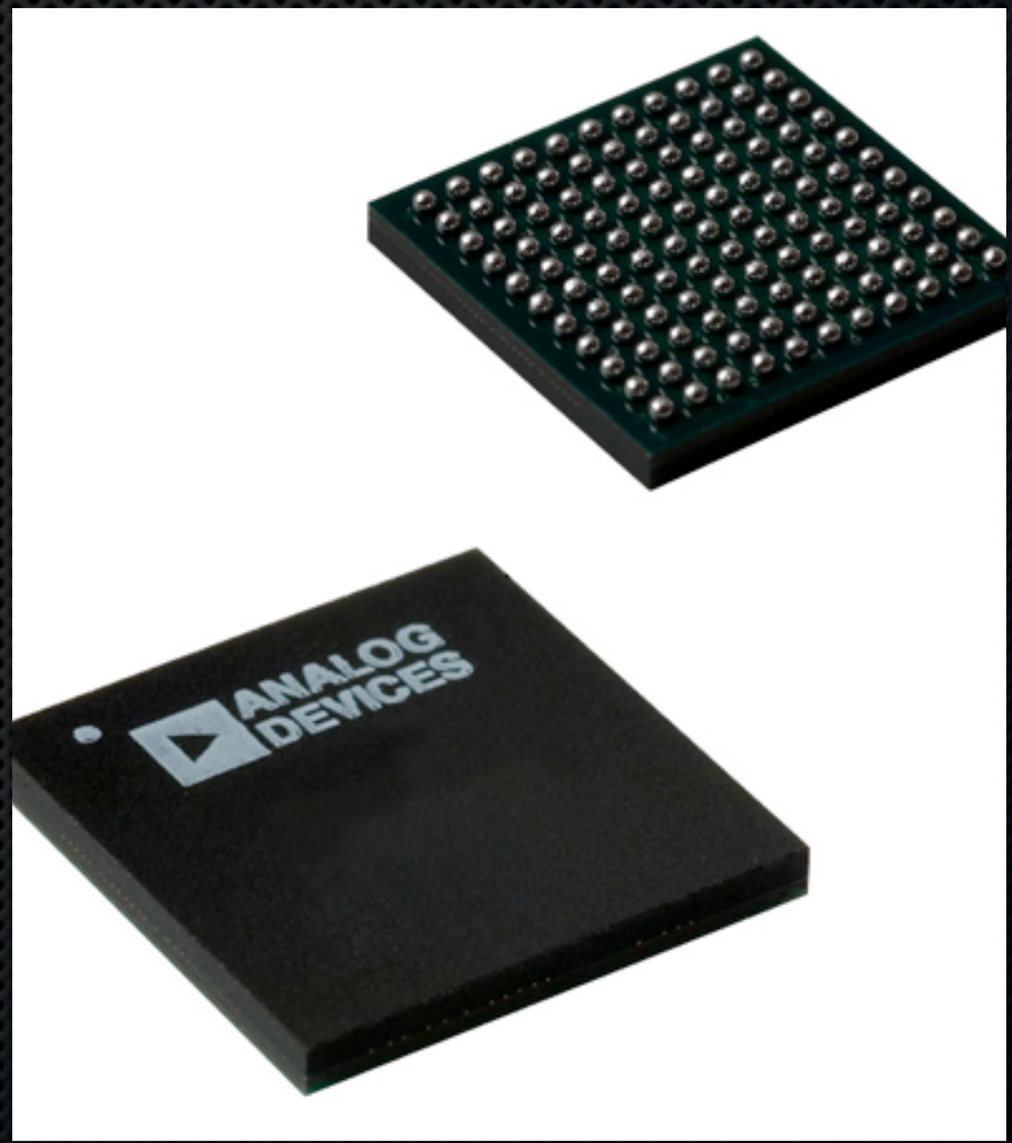


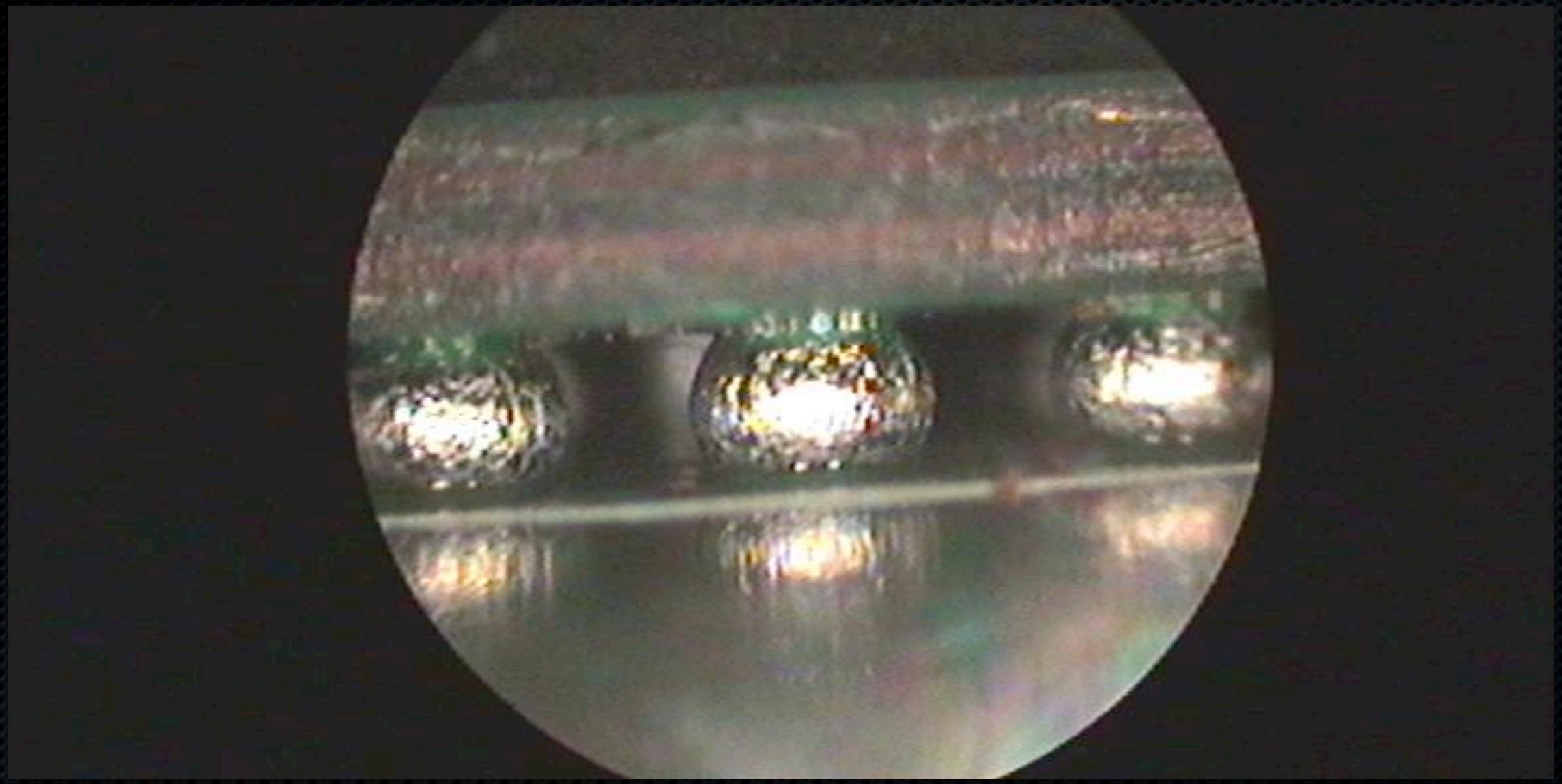
BGA, FPGA og SH4

Rohde, Mads & Brix

Ball Grid Array (BGA)

- SMD-pakke til IC'er
- Kugler af loddetin i stedet for pins
- Ubehagelige at måle på in-circuit

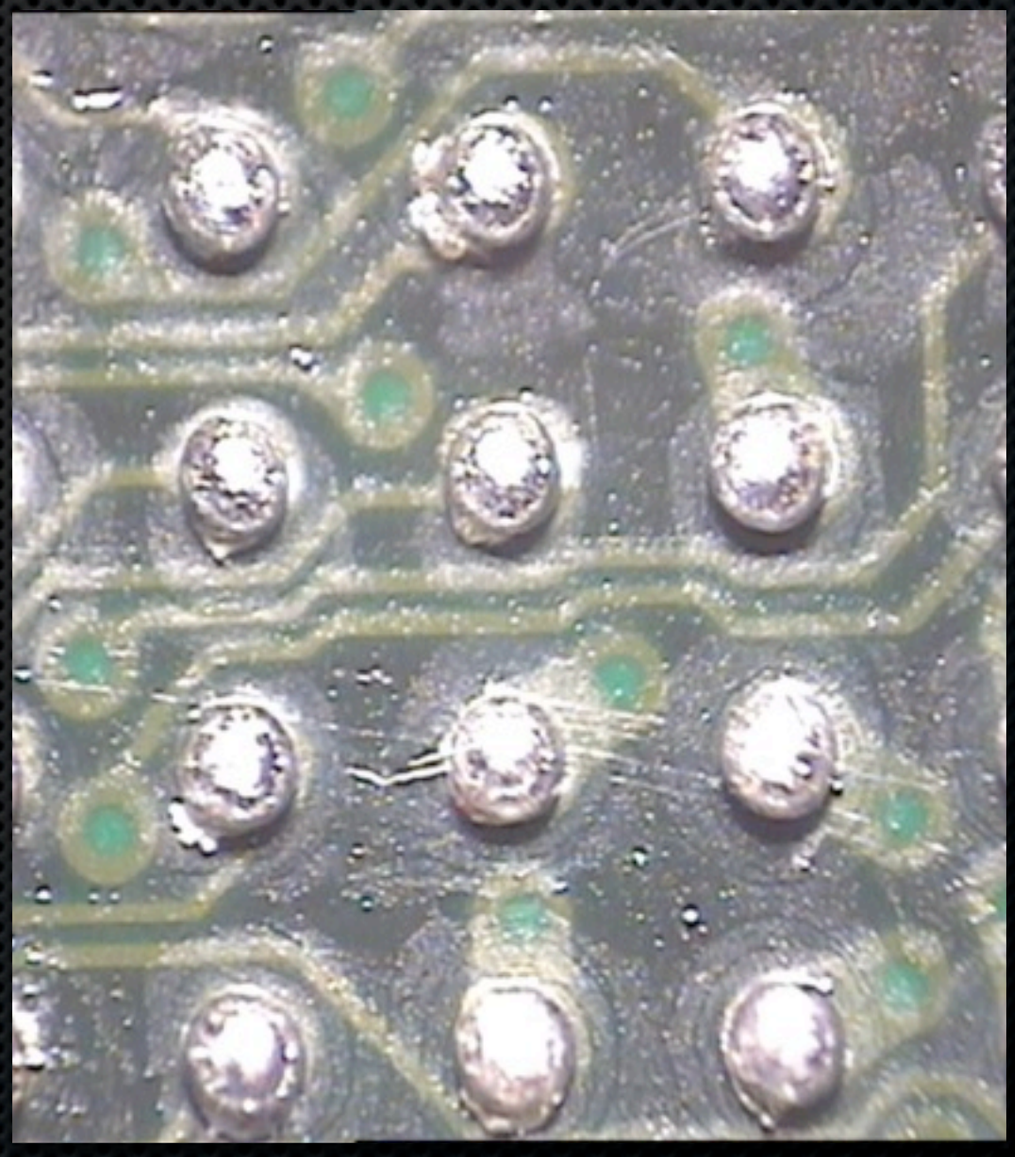


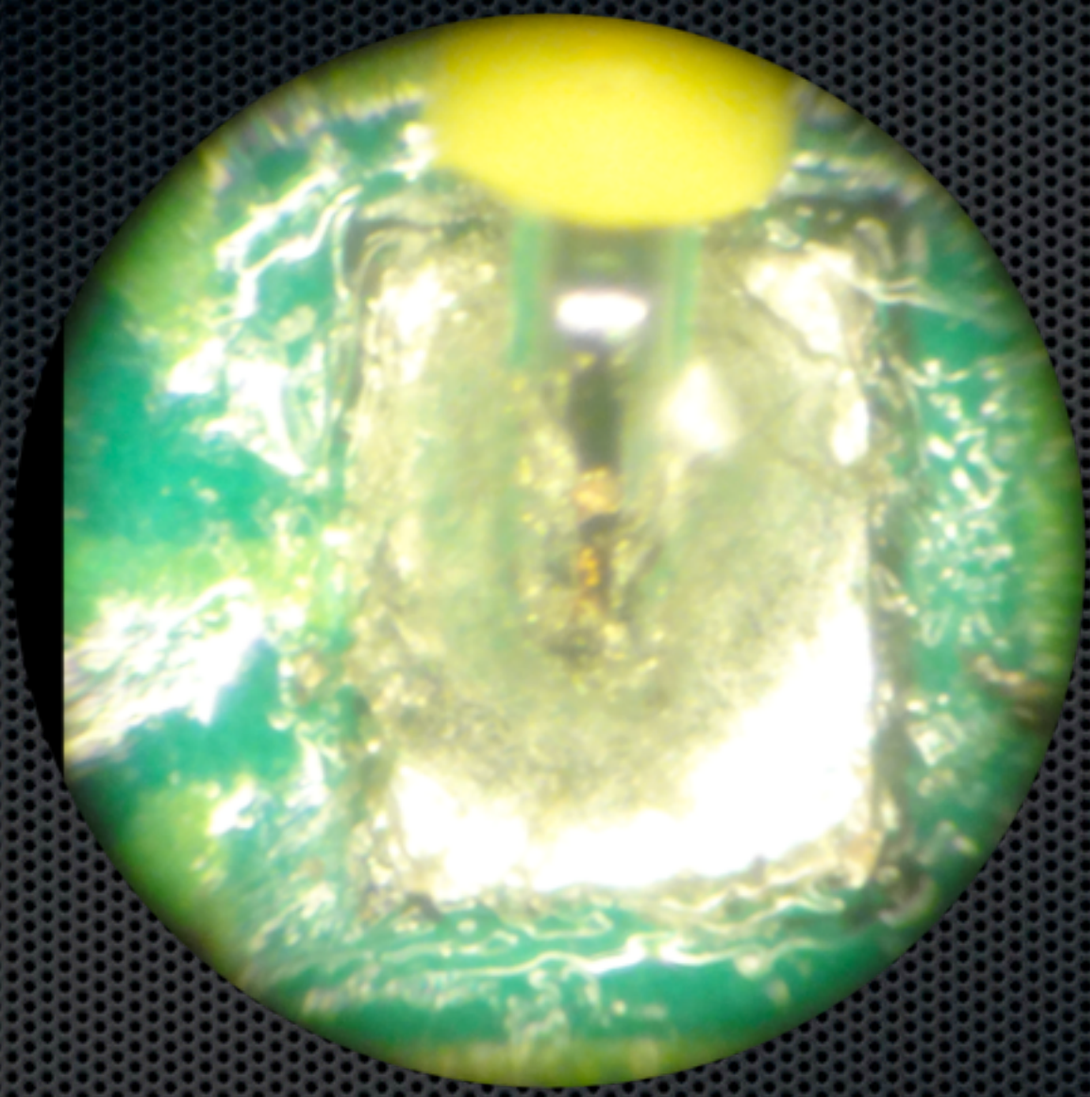


Lodning af BGA

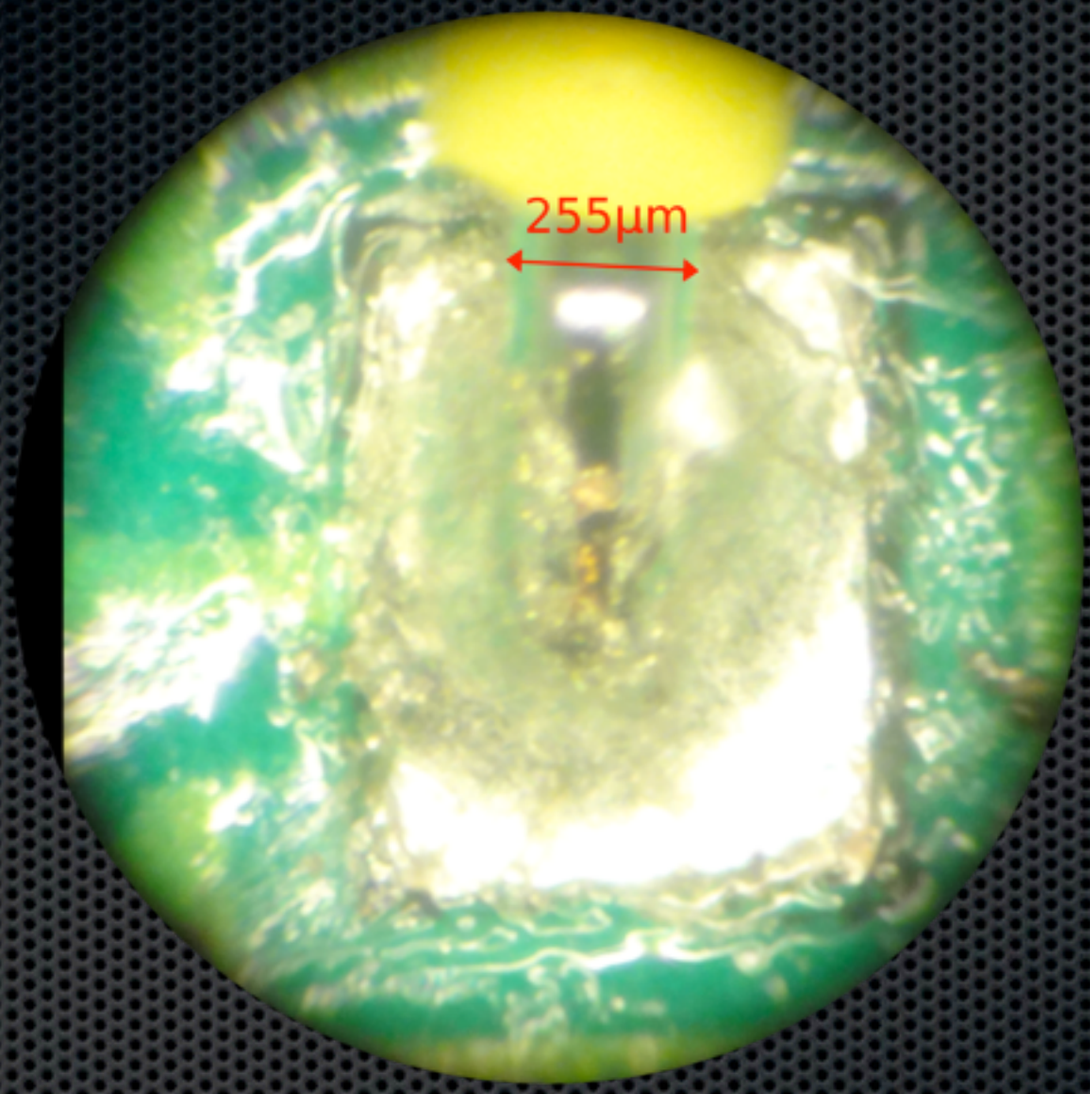
Udlodning af BGA

- Udlodning med infrarød varmekilde
- Automatisk løft af BGA-kreds
- Reballing af BGA-kreds



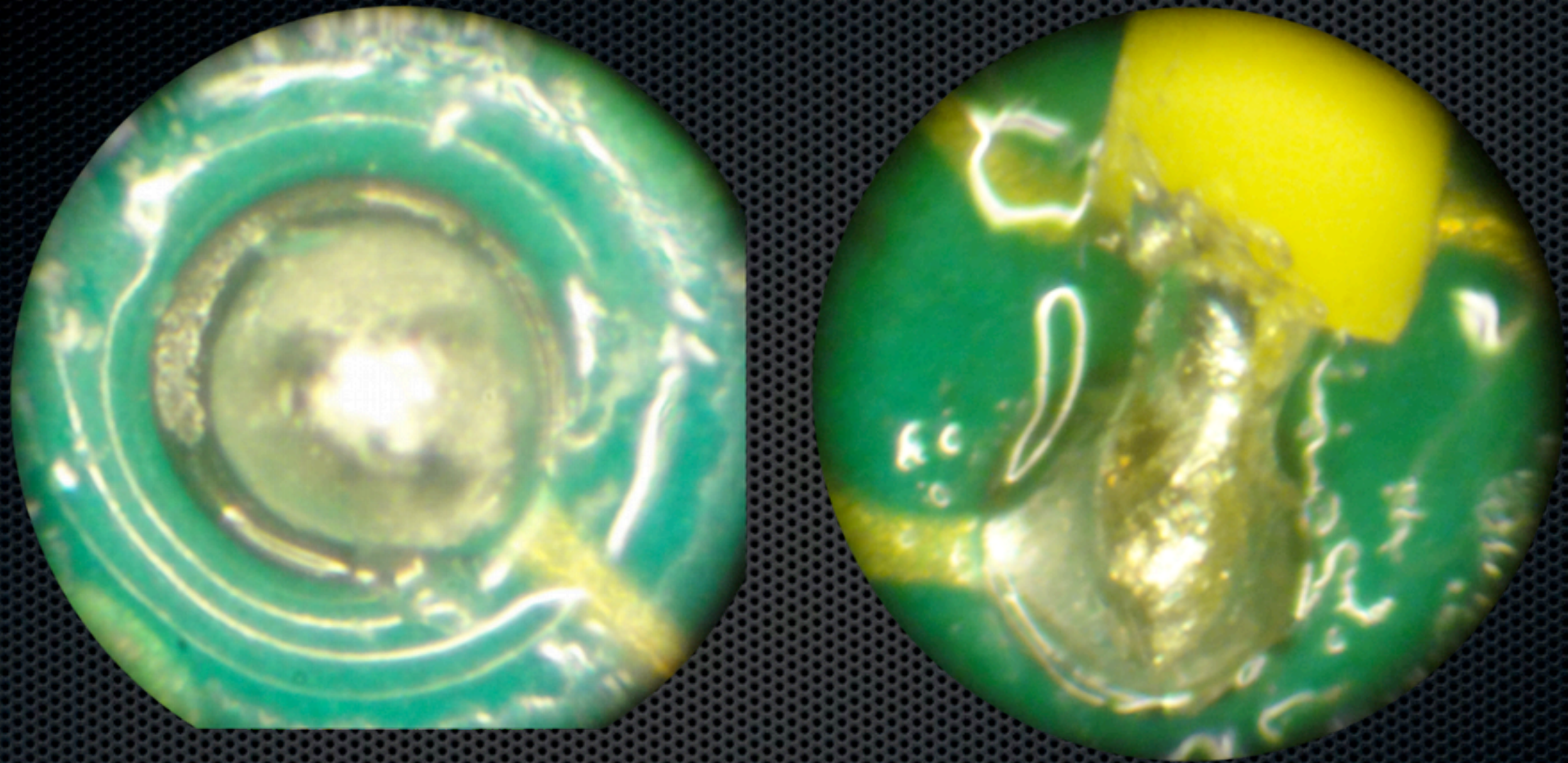


Montage af ledninger



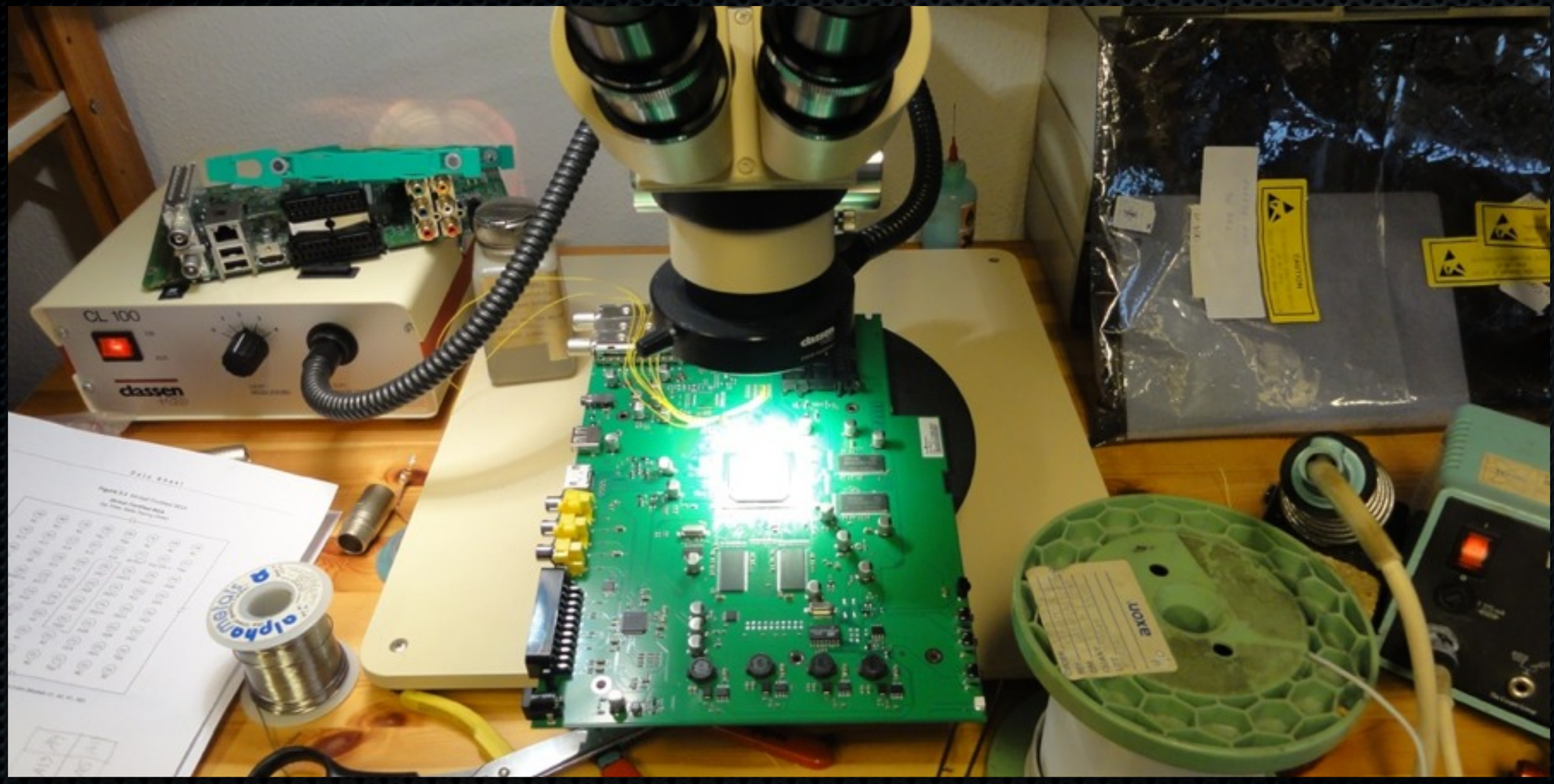
Montage af ledninger

Ledningsdiameter: 255µm (30 AWG)



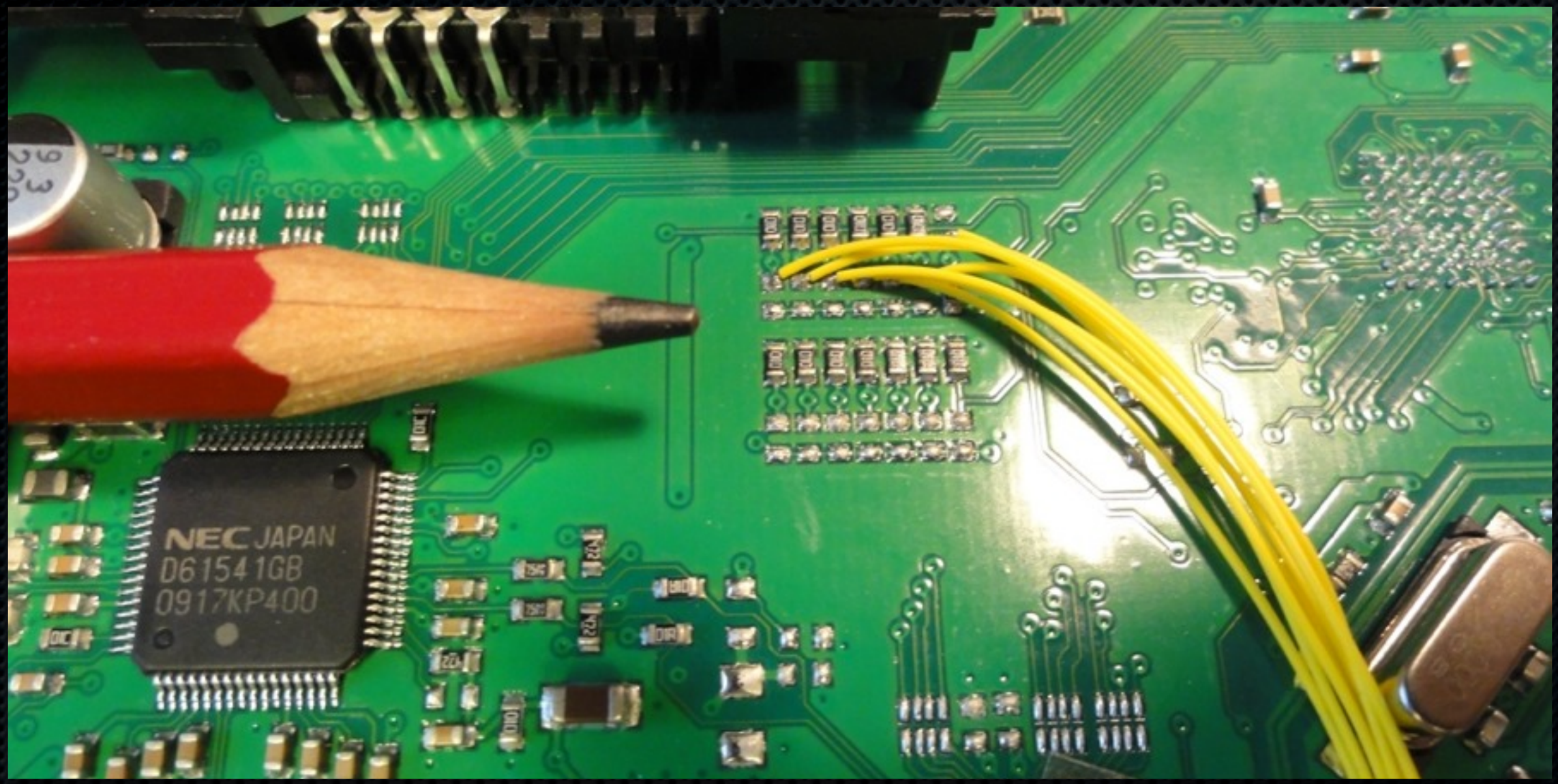
Montage af ledninger

Ledningsdiameter: 255µm (30 AWG)



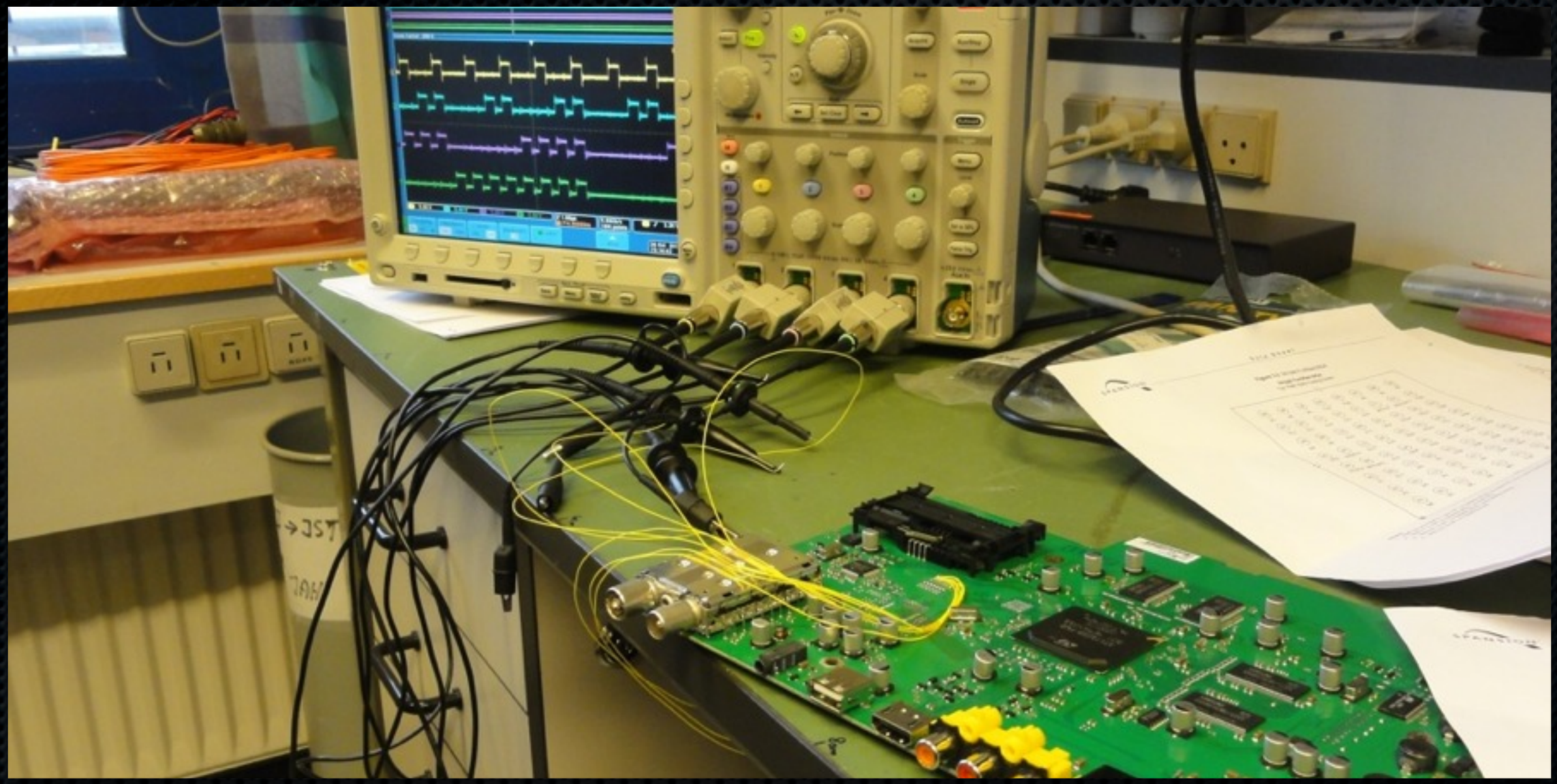
Montage af ledninger

Lodning under mikroskop - ikke vanedannende



Montage af ledninger

8 ledninger loddet på adressebussen

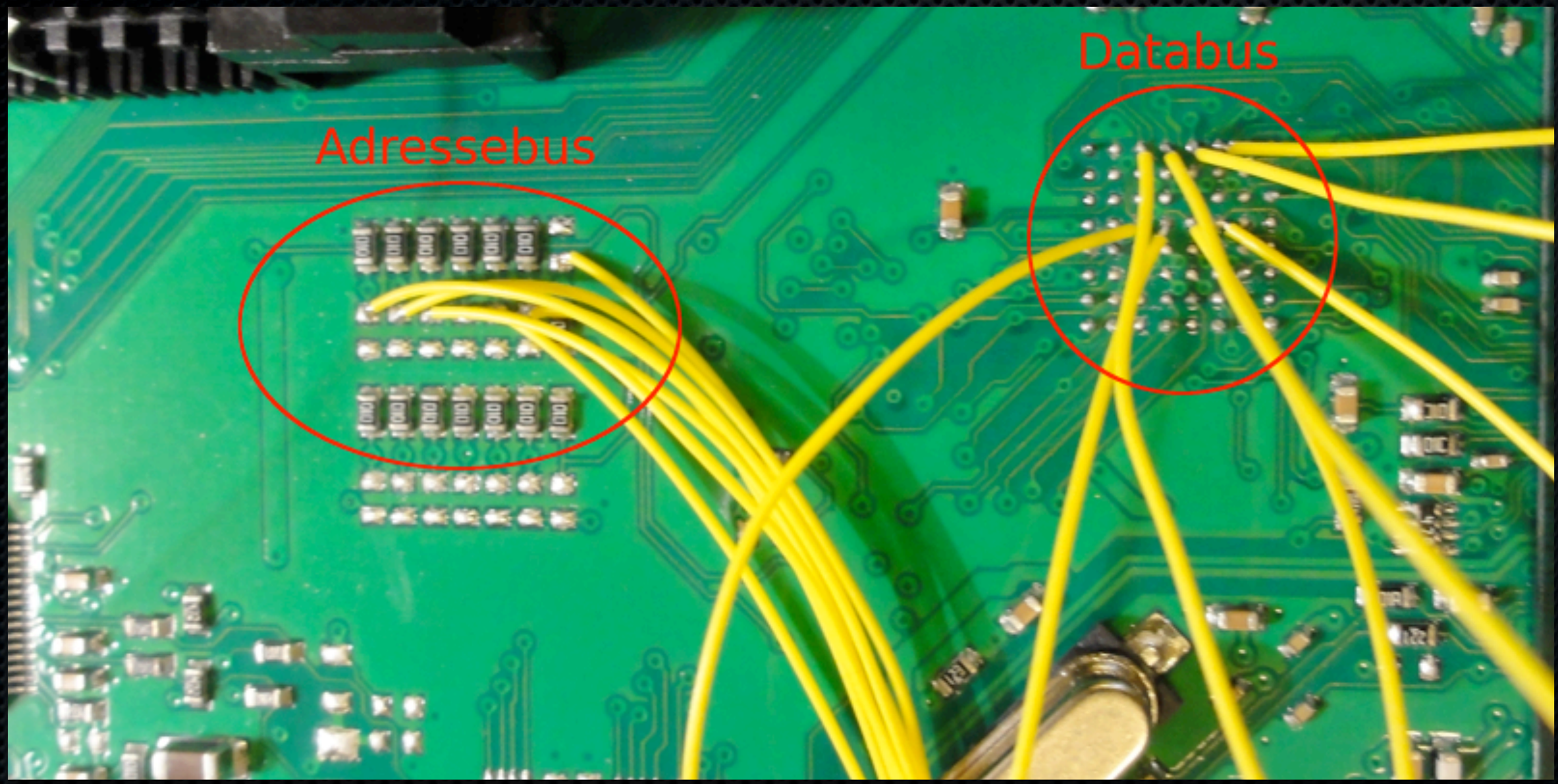


Første testopstilling

Måling på adressebus

Første testopstilling

- 2^0: gul
- 2^1: blå
- 2^2: lilla
- 2^3: grøn



Montage af ledninger

8 ledninger loddet på adressebussen

8 ledninger loddet på databussen

Hurtige loddetips

- Ca. 420°C
- Blyholdigt tin
- Fortinnet spids
- Fikserede ledninger
- Håndledsstøtte
- Tålmodighed!

FPGA Intro

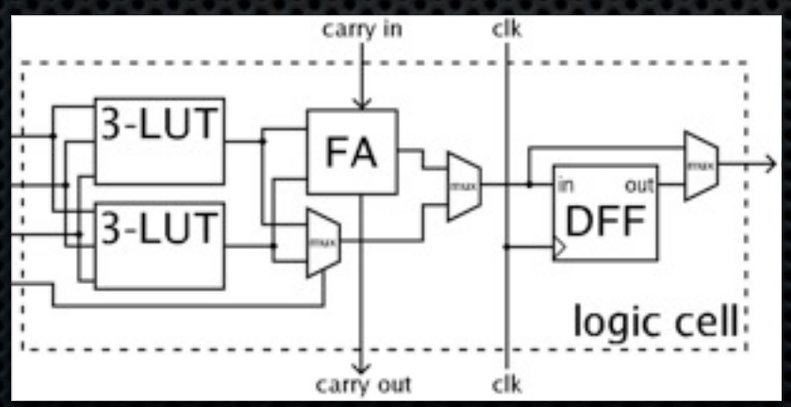
- FPGA: Field Programmable Gate Array
- Reprogrammable logic vs. traditional logic, ASICs and software
- Applications: DSP, Software Defined Radio, CPU emulation,applications requiring real time response and/or applications requiring real time response and/or a large degree of parallelism
- Vendors: 2-3 major players

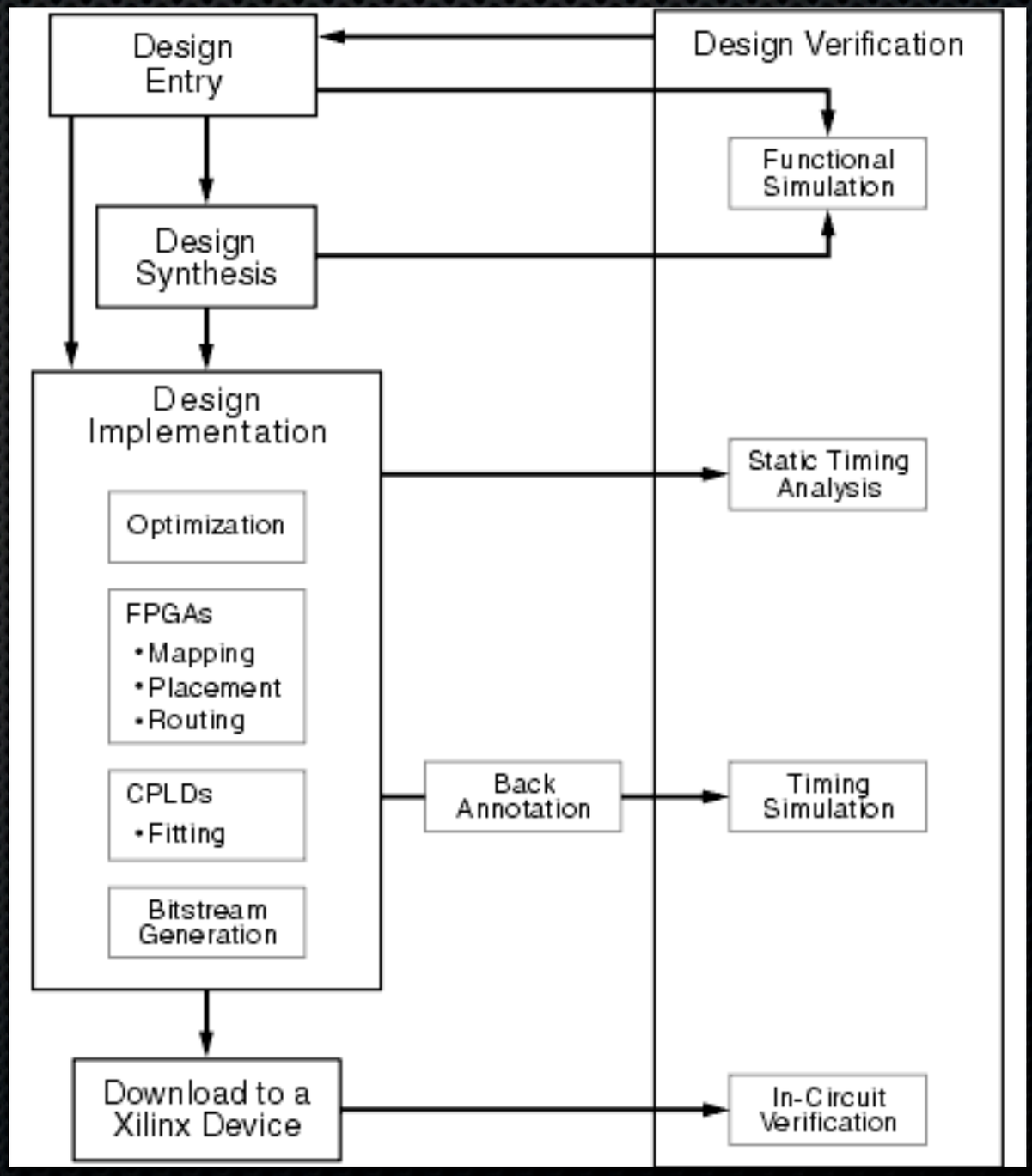
Using FPGAs for prototyping and hacking

- Evaluation boards: Avnet, Trenz Electronic, ...
- Design tools: Xilinx Webpack, Xilinx ISE, Xilinx EDK

FPGA Design Flow

- High level description (VHDL)
- Synthesis
- Place/Route
- Bitstream generation



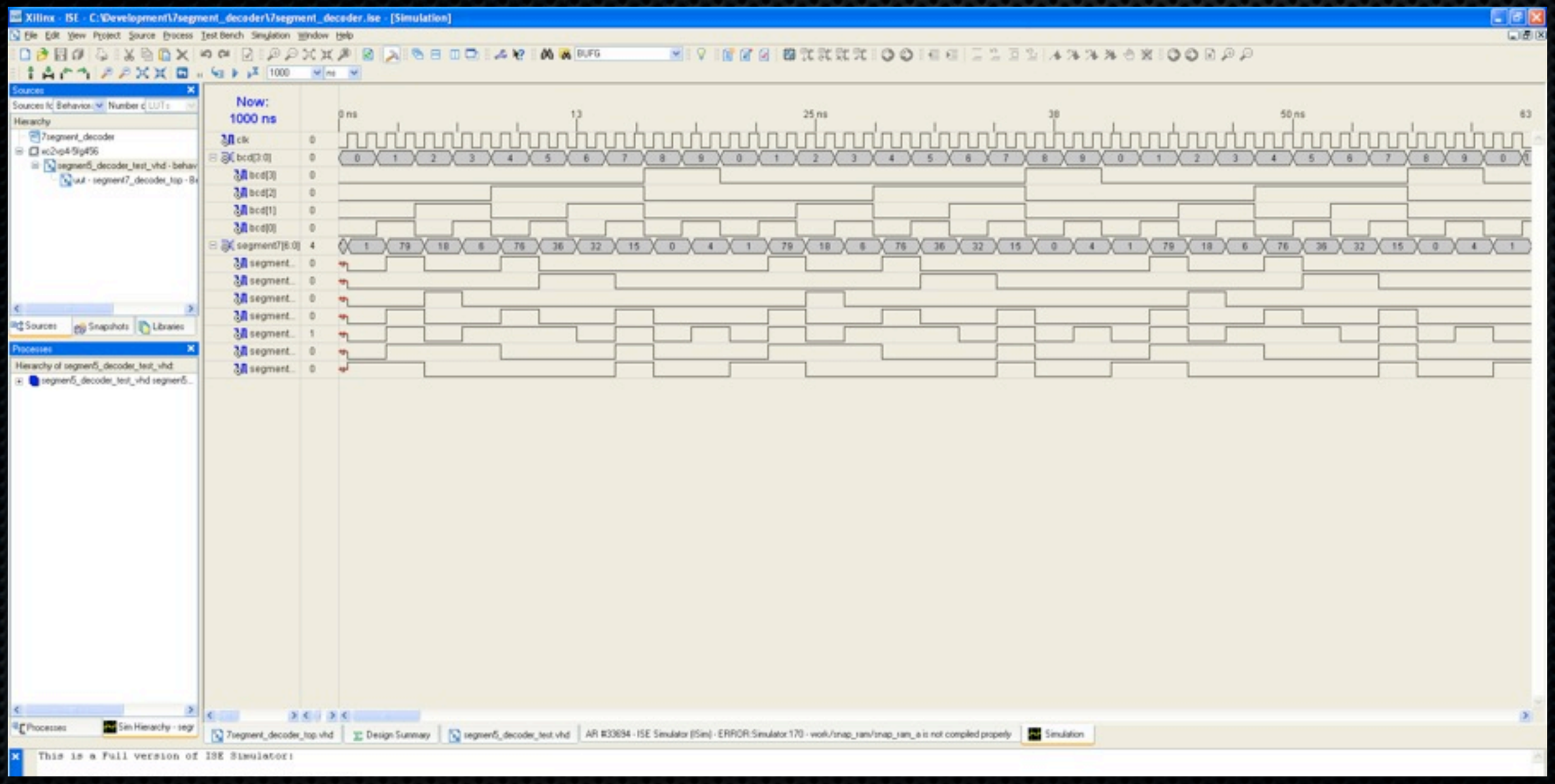


VHDL example - BCD to 7 segment converter

```
----------------------------------------------------------------------------------
-
-- Module Name:    7segment_decoder_top - Behavioral
----------------------------------------------------------------------------------
-
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

---- Uncomment the following library declaration if instantiating
---- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity segment7_decoder_top is
    Port ( clk : in  STD_LOGIC;
           bcd : in  STD_LOGIC_VECTOR (3 downto 0);
           segment7 : out  STD_LOGIC_VECTOR (6 downto 0));
end segment7_decoder_top;

--'a' corresponds to MSB of segment7 and g corresponds to LSB of
segment7.
architecture Behavioral of segment7_decoder_top is

begin

process (clk,bcd)
begin
        if (clk'event and clk='1') then
                case  bcd is
                        when "0000"=> segment7 <="0000001";  -- '0'
                        when "0001"=> segment7 <="1001111";  -- '1'
                        when "0010"=> segment7 <="0010010";  -- '2'
                        when "0011"=> segment7 <="0000110";  -- '3'
                        when "0100"=> segment7 <="1001100";  -- '4'
                        when "0101"=> segment7 <="0100100";  -- '5'
                        when "0110"=> segment7 <="0100000";  -- '6'
                        when "0111"=> segment7 <="0001111";  -- '7'
                        when "1000"=> segment7 <="0000000";  -- '8'
                        when "1001"=> segment7 <="0000100";  -- '9'
                        --nothing is displayed when a number more
than 9 is given as input.
                        when others=> segment7 <="1111111";
                end case;
        end if;
end process;

end Behavioral;
```

VHDL example - testbench

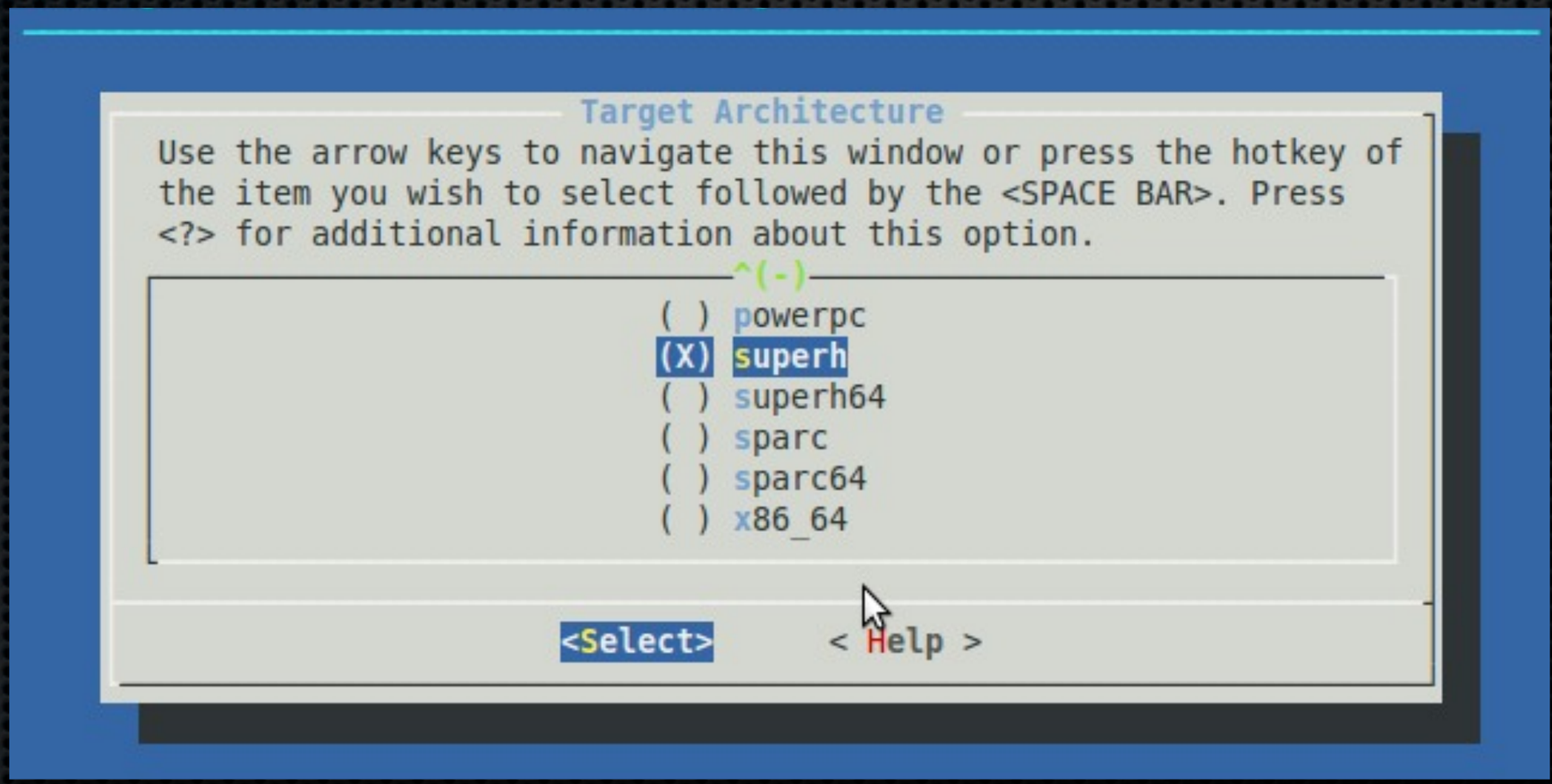
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_unsigned.all;
USE ieee.numeric_std.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;

ENTITY segmen5_decoder_test_vhd IS
END segmen5_decoder_test_vhd;

ARCHITECTURE behavior OF segmen5_decoder_test_vhd IS

        -- Component Declaration for the Unit Under Test (UUT)
        COMPONENT segment7_decoder_top
        PORT(
                clk : IN std_logic;
                bcd : IN std_logic_vector(3 downto 0);
                segment7 : OUT std_logic_vector(6 downto 0)
                );
        END COMPONENT;

        --Inputs
        SIGNAL clk :  std_logic := '0';
        SIGNAL bcd :  std_logic_vector(3 downto 0) := (others=>'0');

        --Outputs
        SIGNAL segment7 :  std_logic_vector(6 downto 0);

  CONSTANT clk_period : time := 1 ns;

BEGIN
        -- Instantiate the Unit Under Test (UUT)
        uut: segment7_decoder_top PORT MAP(
                clk => clk,
                bcd => bcd,
                segment7 => segment7
        );

        tb : PROCESS
        BEGIN

     clk <= '0';
     wait for clk_period/2;
     clk <= '1';
     wait for clk_period/2;

                END PROCESS;

    stim_proc: process
    begin
      for i in 0 to 9 loop
           bcd <= conv_std_logic_vector(i,4);
          wait for 2 ns;
      end loop;
    end process;
END;
```



VHDL example - ModelSim

Build Root

- make menuconfig



- make

SH4 Assembler

```
        .text
        .align 1
        .global  main
        .type    main, @function
main:
        mov.l    uart_baud,r1
        mov      #0x36,r0  /* should be 115200 */
        mov.l    r0,@r1

        mov.w    uart_control_val,r0 /* 8n1, run */
        mov.l    r0,@(0xc,r1)

wait_txbuf:
        mov.l  @(0x14,r1),r0
        and    #0x2,r0
        cmp/eq #0x0,r0    /* if bit 1 is set tx is empty */
        bt         wait_txbuf

        mov    #0x5a,r0
        mov.l  r0,@(0x4,r1)

        mov.l  rx_byte_addr,r2
        jsr    @r2
        nop

        bra    main
rx_byte:
        mov.l  @(0x14,r1),r0
        and    #0x1,r0
        cmp/eq #0x0,r0
        bt     rx_byte

        rts
        mov.l  @(0x8,r1),r0

uart_control_val: .short 0x489
uart_baud:        .long 0xb803000
rx_byte_addr: .long rx_byte
/* uart_control:    .long 0xb80300c */
/* uart_tx:         .long 0xb803004 */
/* uart_rx:     .long 0xb803008 */
/* uart_status:  .long 0xb803014 */

/* Some memory at 0x7c00_1000 */

        .size     main, .-main
```

SH4 Linker

```
SECTIONS
{
  . = 0;
  .text : { *(.text) }
  .data : { *(.data) }
  .bss : { *(.bss) }
}
```

SH4 Makefile

```
CROSS_COMPILE=/home/rohde/develop/buildroot/build_sh4/staging_dir/usr/bin/sh4-linux-
CC=$(CROSS_COMPILE)gcc
AS=$(CROSS_COMPILE)as

bl.txt: bl.o
	$(CC) -nostdlib -Wl,-T link.ld bl.o -o bl
	$(CROSS_COMPILE)objcopy -O binary bl bl.bin
	./make_vhdl <bl.bin>bl.txt
```