Tämä sivu toimii Digitaalitekniikka-kurssin leikepöytänä

- https://kirjoitusalusta.fi/hacklab-kurssi-digitaalitekniikka
- https://github.com/HelsinkiHacklab/digitaalitekniikka

Torstai 17.5.

- suovula: ZCU Aurora SFP-link
- Jari: OPL3-FPGA - https://github.com/gtaylormb/opl3_fpga
    - I2S testi -> OK
    - rekisteribankin mappaus emulaattoriin?
- Ossi: custom floating point
    - FPGA Prototyping by Verilog Examples□Xilinx Spartan-3 Version (2008).pdf
        - 3.9.2 Sign-magnitude adder -> OK
        - 3.9.4 Simplified floating-point adder -> OK

Torstai 3.5.

- suovula: Ubuntu 18.04 32/64-bit Zynq distrot
- Kremmen: XADC projektin kääntäminen
- Timo: LVDS-adapterin ja kaapelin testaus Z-turnilla ->OK
- Antti: ?
- Jari: OPL3-FPGA
    - I2S testi?
        - https://store.digilentinc.com/pmod-i2s-stereo-audio-output/
    - https://github.com/gtaylormb/opl3_fpga
- Ossi: custom floating point
    - FPGA Prototyping by Verilog Examples□Xilinx Spartan-3 Version (2008).pdf
        - 3.9.2 Sign-magnitude adder -> OK
        - 3.9.4 Simplified floating-point adder -> OK
        -

Torstai 26.4.

- suovula: ZCU120 smoketest -> OK
- Kremmen: mittasauvan signaalit skoopilla
    - ~3V eli vastusjakoa tarvitaan?
- Timo: LVDS-adapterin ja kaapelin testaus Z-turnilla ->OK
- Antti: ?
- Jari: OPL3-FPGA
    - I2S testi?
        - https://store.digilentinc.com/pmod-i2s-stereo-audio-output/
    - https://github.com/gtaylormb/opl3_fpga
- Ossi: custom floating point
    - FPGA Prototyping by Verilog Examples□Xilinx Spartan-3 Version (2008).pdf
        - 3.9.2 Sign-magnitude adder -> OK
        - 3.9.4 Simplified floating-point adder -> OK

Torstai 5.4.

- suovula: 7z030-piirin GTX-transceiverit SFP+ 6.6Gbps loopback testaus
- Kremmen: mittasauvan 1v signaalit skoopilla
  - etsii speksiä?
- Timo: LVDS-adapterin ja kaapelin testaus Z-turnilla?
- Antti: PDP-8 emulaattori
  - käy läpi turpo_cpu.v koodin
    - https://github.com/HelsinkiHacklab/digitaalitekniikka/tree/master/tutorials/12-turpo-cpu
  - NAND2Tetris
- Jari: OPL3-FPGA
  - I2S testi?
    - https://store.digilentinc.com/pmod-i2s-stereo-audio-output/
  - https://github.com/gtaylormb/opl3_fpga
- Ossi: custom floating point?
  - FPGA Prototyping by Verilog Examples□Xilinx Spartan-3 Version (2008).pdf
    - 3.9.2 Sign-magnitude adder
    - 3.9.4 Simplified floating-point adder

Torstai 15.3.

- https://github.com/pervognsen/bitwise/blob/master/README.md

Torstai 8.3.

- Buildroot BusyBox -linux QSPI-flashille
  - hae uudet boottitiedostot githubista: https://github.com/HelsinkiHacklab/digitaalitekniikka/tree/master/boards/boot_files
    - **uImage** ja **rootfs.cpio.uboot** - kaikille yhteisiä
    - [**uEnv.txt** - muista editoida MAC osoitetta ethaddr!]
    - **boot.bin** ja **devicetree.dtb** - levykohtaisia
  - serial portin kautta terminaaliyhteys levylle
    - export TERM=xterm-256color && resize
    - sudo mkdir -p /mnt/fat && sudo mount -o rw /dev/mmcblk0p1 /mnt/fat
      - cd /mnt/fat
      - df -h .
  - bootataan u-boot promptiin: **zybuntu-u-boot>**
    - sf probe 0 100000000 0
      - varmuuskopio vanhasta flashistä:
      - sf read 0x00042000 0x0000000 0x1000000
        - fatwrite mmc 0 0x00042000 zturn-qspi.img 0x1000000
    - **boot.bin** (step-by-step)
      - fatload mmc 0 0x00042000 boot.bin
      - sf erase 0x0000000 +${filesize}
      - sf write 0x00042000 0x0000000 ${filesize}

        - **uImage** (kaikki yhdellä rivillä)
            - fatload mmc 0 0x00042000 uImage && sf erase 0x0100000 +${filesize} && sf write 0x00042000 0x0100000 ${filesize}
        - **devicetree.dtb**
            - fatload mmc 0 0x00042000 devicetree.dtb && sf erase 0x0600000 +${filesize} && sf write 0x00042000 0x0600000 ${filesize}
        - **rootfs.cpio.uboot**
            - fatload mmc 0 0x00042000 rootfs.cpio.uboot && sf erase 0x0620000 +${filesize} && sf write 0x00042000 0x0620000 ${filesize}

Torstai 22.2.

- ToF-kamerat
    -
- piirilevyt
    - BGA-breakout?
        - http://www.vlsi.fi/fileadmin/datasheets/vs23s040.pdf
    - stencilit?
        - https://www.youtube.com/watch?v=JWUJtmgh55M
- BusyBox / vastaavat
    - rootfs.cpio.uboot
        - https://www.kernel.org/doc/Documentation/filesystems/ramfs-rootfs-initramfs.txt
    - https://buildroot.org/
    -

Torstai 15.2.

- CML->LVDS-lautakolehti
    - suovula: 20e
    - antti: 10e
    - timo: 14.5e
    - knorad: 18.5 €
    - jari: 20e
    - thjt: 20e
    - janne 10 e
    - torsti 15e
    - Yliolli 20e

Torstai 11.12.

- mountataan USB-tikku
    - sudo mkdir -p /mnt/usb && sudo mount -o rw /dev/sda1 /mnt/usb
- linkkejä:

- open source toolchain Latticen tietyille piireille
  - https://www.youtube.com/watch?v=8OaZ89TN0fo
- videomoodien ajastuksia
  - https://www.mythtv.org/wiki/Modeline_Database
- extratehtäviä:
  - toteuta ***void put_pixel(int32_t x, int32_t y, uint32_t color);***
    - globaalit vakiot ***WIDTH***, ***HEIGHT***
    - globaali taulukko ***uint32_t video_mem[WIDTH*HEIGHT];***
  - seuraavan voi tehdä työpöytälinuksilla tai vähän hitaammin zynq-laudallakin:
  - asenna **youtube-dl**
    - sudo apt-get install youtube-dl
    - lataa joku 3-6min video youtubesta, esim:
      - **youtube-dl** https://www.youtube.com/watch?v=Q0CbN8sfihY
      - **mv** "Star Wars - The Last Jedi Trailer (Official)-Q0CbN8sfihY.mkv" **starwars.mkv**
  - asenna **ffmpeg** työkalukokelma
    - sudo apt-get install ffmpeg
    - tee lataamastasi videosta 32-bittinen 400x300 resoluution raakaversio, vaatii gigoittain tilaa
      - ffmpeg -i "**starwars.mkv**" -f rawvideo -pix_fmt bgra -s 400x300 -vcodec rawvideo "**starwars-400x300.bgra**"
    - kopioi se USB-tikulle ja mountaa zybuntussa **/mnt/usb**

Torstai 18.1.

- pinnimappauksia
  - https://www.xilinx.com/support/packagefiles/z7packages/xc7z010clg400pkg.txt
  - https://www.xilinx.com/support/packagefiles/z7packages/xc7z020clg400pkg.txt

Torstai 4.12.

- mitä tehdään keväällä?
  - LCD, HDMI?
    - näytonohjain
  - Hall-anturi-resolveri?
  - Linux järjestelmän kasaus, BusyBox?
  - SystemVerilog vs. HLS?
  - etäisyysmittaus
    - TOF
    - FMCW
    - Phase-Shift Laser Range-Finding
  - SDR

Torstai 26.10.

- asennetaan https://processing.org/
- https://github.com/HelsinkiHacklab/digitaalitekniikka/tree/master/tutorials/7-axis-xadc
- axis_server.c:
  - https://raw.githubusercontent.com/HelsinkiHacklab/digitaalitekniikka/master/tutorials/7-axis-xadc/axis_server.c
- ProcessingTest.pde:
  - https://raw.githubusercontent.com/HelsinkiHacklab/digitaalitekniikka/master/tutorials/7-axis-xadc/ProcessingTest.pde
- Lineaarinen HALL-anturi 3V3
  - https://www.digikey.com/products/en?keywords=EQ-731L-ND

Torstai 12.10.

- Arduino + Processing Demo
- XADC -> AXI-Stream

# fifo fill
sudo busybox devmem 0x43c00008

# fifo data
sudo busybox devmem 0x43c0000c

XADC - pynq
set_property -dict { PACKAGE_PIN D18   IOSTANDARD LVCMOS33 } [get_ports { A0_N }]; #IO_L3N_T0_DQS_AD1N_35 Sch=ck_an_n[0]
set_property -dict { PACKAGE_PIN E17   IOSTANDARD LVCMOS33 } [get_ports { A0_P }]; #IO_L3P_T0_DQS_AD1P_35 Sch=ck_an_p[0]

XADC - zturn
set_property -dict { PACKAGE_PIN B20   IOSTANDARD LVCMOS33 } [get_ports { A0_N }]; # IO_L1N_T0_AD0N_35
set_property -dict { PACKAGE_PIN C20   IOSTANDARD LVCMOS33 } [get_ports { A0_P }]; # IO_L1P_T0_AD0P_35

Torstai 5.10.

- korjaa **sudo**-komennon TAB-täydennyksen
  - sudo apt-get install --reinstall bash-completion
- **devmem**-komento zybuntuun

        - sudo ln -s /bin/busybox /bin/devmem
- button debounce
    - https://hackaday.com/2015/12/09/embed-with-elliot-debounce-your-noisy-buttons-part-i/

Torstai 28.9.

- jatketaan AXI4-Stream lohkojen liimailuja
- kopio USB-tikulta esimerkki .raw-tiedosto Zynqille
    - mountataan USB-tikku
        - sudo mkdir -p /mnt/usb && sudo mount -o rw /dev/sda1 /mnt/usb
    - mennään projektikansioon, esim.
        - cd ~/workspace/dac_test/
        - cp /mnt/usb/*.raw .
    - unmountata (~eject)
        - umount /mnt/usb
- **axilite_axis**-lohkon testaus loopback-moodissa
    - output-fifossa tilaa?
        - sudo busybox devmem 0x43c00000 32
    - input-fifossa dataa?
        - sudo busybox devmem 0x43c00008 32

    - output-fifoon kirjoitus:
        - sudo busybox devmem 0x43c00004 32 0x12345678
    - input-fifosta data:
        - sudo busybox devmem 0x43c0000c 32

Torstai 21.9.

- sudo nano /etc/udev/rules.d/xdevcfg.rules
    - KERNEL=="xdevcfg", GROUP="root", MODE="0666"
    - vastaa samaa kuin
    - sudo chmod 666 /dev/xdevcfg
- kerrataan AXI4-Stream protokolla

- ffmpeg -i testi.mp4/.mp3/.flac -f s16le -ar 44100 -acodec pcm_s16le testi-44100-s16le.raw
    - asennus omalle zynqille jos haluaa: sudo apt-get install ffmpeg

Torstai 14.9.

- uusi kerneli 4.9.0
    - https://github.com/HelsinkiHacklab/digitaalitekniikka/tree/master/boards/boupot_files
- mittaillain WiFi-tikkujen virrankulutus
- tutustutaan **connmanctl**-komentoon
    - https://wiki.archlinux.org/index.php/ConnMan

    - https://gist.github.com/kylemanna/6930087
- valmistaudutaan ensimmäiseen varsinaiseen Zynq-projektiin
    - kerrataan Verilog syntaksi, kombinaatio- ja sekvenssilohkot
    - katsotaan miten kehityslaudoilta saa audiota ulos 3.5mm plugiin
        - pynq sisältää valmiina mono-outputin, mutta voidaan tehdä oma stereo-liitäntä kaikille levyille ensi viikoksi
- tehtiin AXI4-Lite kertaus
    - https://github.com/HelsinkiHacklab/digitaalitekniikka/tree/master/tutorials/5-axi-led

Torstai 7.9.

- New board files, kopio **.../Xilinx/Vivado/2016.4/data/boards/board_files/**
    - https://github.com/HelsinkiHacklab/digitaalitekniikka
    - Nämä kopioidan board_files-kansioon Vivadon alle, esimerkiksi: .../**Xilinx/Vivado/2016.4/data/boards/board_files/**
- microSD-kortin FAT-partiton mounttaaminen
    - sudo mkdir -p /mnt/fat && sudo mount -o rw /dev/mmcblk0p1 /mnt/fat
- USB-tikun mounttaaminen
    - sudo mkdir -p /mnt/usb && sudo mount -o rw /dev/sda1 /mnt/usb

Torstai 31.8.

- Xilinx - Zynq Technical Reference Manual - UG585 (2016-09-27)
    - https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf
- Xilinx - 7 Series Libraries Guide - UG953 (2016-11-30)
    - https://www.xilinx.com/support/documentation/sw_manuals/xilinx2017_2/ug953-vivado-7series-libraries.pdf

-
-
-
- ++
- scp tutorial-4-fclk-led-blink.runs/impl_1/design_1_wrapper.bit user@zybuntu:/dev/xdevcfg
-
- **sudo fdisk /dev/mmcblk0**
-
-

Switches
set_property -dict { PACKAGE_PIN R19   IOSTANDARD LVCMOS33 }   [get_ports { BUTTON[0] }];   # IO_0_34

set_property -dict { PACKAGE_PIN T19   IOSTANDARD LVCMOS33 }   [get_ports { BUTTON[1] }];   # IO_25_34
set_property -dict { PACKAGE_PIN G14   IOSTANDARD LVCMOS33 }   [get_ports { BUTTON[2] }];   # IO_0_35
set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMOS33 }   [get_ports { BUTTON[3] }];   # IO_25_35

RGB LEDs
set_property -dict { PACKAGE_PIN R14   IOSTANDARD LVCMOS33 }   [get_ports { LED[0] }];   # IO_L6N_T0_VREF_34
set_property -dict { PACKAGE_PIN Y16   IOSTANDARD LVCMOS33 }   [get_ports { LED[1] }];   # IO_L7P_T1_34
set_property -dict { PACKAGE_PIN Y17   IOSTANDARD LVCMOS33 }   [get_ports { LED[2] }];   # IO_L7N_T1_34
set_property -dict { PACKAGE_PIN W13   IOSTANDARD LVCMOS33 }   [get_ports { LED[3] }];   # IO_L4N_T0_34   J5.8

Torstai 8.6.

AXI-Lite

https://github.com/HelsinkiHacklab/digitaalitekniikka/tree/master/tutorials/3-axilite

Zynq AXI tutoriaaleja

- https://www.youtube.com/watch?v=nAycgPUOiAI&index=3&list=PL4D6H9w4Ihdp0YRExoQXfbVMRL1GgEPCm&t=18s
- https://www.youtube.com/watch?v=_F124UaZ-d0

Torstai 1.6.

https://github.com/HelsinkiHacklab/digitaalitekniikka/tree/master/tutorials/3-axilite

ARM / AMBA / AXI4-protokollat ('*väylät*')

- https://www.xilinx.com/support/documentation/ip_documentation/ug761_axi_reference_guide.pdf
- AXI4 ('*Full*')
    - bursts
- AXI4-Lite
    - single beats
    - 32-bit addresses, 32/64-bit data
- AXI4-Stream
    - M_AXIS -> S_AXIS
    - M_AXIS -> S_AXIS_S2MM

Torstai 18.5.

```
// tutorial_top_4.v - Simple 'Larson scanner'

`timescale 1ps / 1ps
`default_nettype none

module tutorial_top #(
      parameter              FCLK0_FREQ = 100_000_000
   ) (
      input  wire             FCLK0,
      input  wire [4-1:0]        BUTTON,
      output reg  [4-1:0]        LED
   );

   wire                      clk;
   IBUFG ibuf_clk(
      .I(FCLK0),
      .O(clk)
   );

   reg  [32-1:0]              scaler = 'd0;

   reg  [3-1:0]              counter = 3'b000;

   // Sequential logic
   always @(posedge FCLK0) begin
      if (scaler == 'd0) begin
         scaler <= FCLK0_FREQ/'d2 - 'd1;

         // Transfer next state to register
         if (counter < 'd5) begin
            counter <= counter + 'd1;
         end else begin
            counter <= 'd0;
         end

      end else begin
```

```
            scaler <= scaler - 'd1;
        end

        case (counter)
            3'd0 : LED <= 4'b0001;
            3'd1 : LED <= 4'b0010;
            3'd2 : LED <= 4'b0100;
            3'd3 : LED <= 4'b1000;
            3'd4 : LED <= 4'b0100;
            3'd5 : LED <= 4'b0010;
            default: LED <= 4'b1111;
        endcase

    end

endmodule
```

--

```
// tutorial_top_3.v - Simple 'Larson scanner'
timescale 1ps / 1ps
`default_nettype none

module tutorial_top #(
        parameter                   FCLK0_FREQ = 100_000_000
    ) (
        input  wire                 FCLK0,
        input  wire [4-1:0]         BUTTON,
        output reg  [4-1:0]         LED
    );

    wire                            clk;
    IBUFG ibuf_clk(
        .I(FCLK0),
        .O(clk)
    );

    reg  [32-1:0]                   scaler = 'd0;

    reg  [3-1:0]                    counter_next;
    reg  [3-1:0]                    counter_reg = 3'b000;

    // Sequential logic
    always @(posedge FCLK0) begin
        if (scaler == 'd0) begin
            scaler <= FCLK0_FREQ/'d2 - 'd1;

            // Transfer next state to register
            counter_reg <= counter_next;

        end else begin
```

`

```
            scaler <= scaler - 'd1;
        end
    end

    // Next state
    always @* begin
        if (counter_reg < 'd5) begin
            counter_next = counter_reg + 'd1;
        end else begin
            counter_next = 'd0;
        end
    end

    // Output logic
    always @* begin
        case (counter_next)
            3'd0 : LED = 4'b0001;
            3'd1 : LED = 4'b0010;
            3'd2 : LED = 4'b0100;
            3'd3 : LED = 4'b1000;
            3'd4 : LED = 4'b0100;
            3'd5 : LED = 4'b0010;
            default: LED = 4'b1111;
        endcase
    end

endmodule
```

--

Verilog : https://discourse.hacklab.fi/t/kurssi-digitaalitekniikka-torstaisin-kello-18-00/540/18

Larson skanner, ota pohjaksi vaikka
https://github.com/HelsinkiHacklab/digitaalitekniikka/tree/master/tutorials/2-sequential-logic

Ubuntu + Vivado 2017.1 bugi:

- Vivado 2017.1 java.lang.UnsupportedOperationException: This is supposed to be overridden by subclasses.
- https://www.xilinx.com/support/answers/69088.html

Torstai 11.5.

// tutorial_top_2.v - Simple 'Larson scanner'

```
`timescale 1ps / 1ps
`default_nettype none

module tutorial_top #(
        parameter                   FCLK0_FREQ = 100000000
    ) (
        input  wire                 FCLK0,
        input  wire [4-1:0]         BUTTON,
        output wire [4-1:0]         LED
    );

    wire                        clk;
    IBUFG ibuf_clk(
        .I(FCLK0),
        .O(clk)
    );

/*
    reg  [32-1:0]               scaler_next;
    reg  [32-1:0]               scaler_reg = 'd0;

    always @* begin
        if (scaler_reg == 'd0) begin
            scaler_next = FCLK0_FREQ - 'd1;
        end else begin
            scaler_next = scaler_reg - 'd1;
        end
    end

    always @(posedge clk) begin
        scaler_reg <= scaler_next;
    end
*/

    reg  [32-1:0]               scaler = 'd0;

    always @(posedge clk) begin
        if (scaler == 'd0) begin
            scaler <= FCLK0_FREQ - 'd1;
        end else begin
            scaler <= scaler - 'd1;
        end
    end


    /*
    reg  [5-1:0]                larson_next;
    reg  [5-1:0]                larson_reg = 5'b0_0001;     // Register, the R in RTL

    // Combinatorial logic, the L in RTL
    always @* begin
        larson_next = 5'b0_0001;
```

```
        case (larson_reg)
            5'b0_0001 : larson_next = 5'b0_0010;
            5'b0_0010 : larson_next = 5'b0_0100;
            5'b0_0100 : larson_next = 5'b1_1000;
            5'b1_1000 : larson_next = 5'b1_0100;
            5'b1_0100 : larson_next = 5'b1_0010;
            5'b1_0010 : larson_next = 5'b0_0001;
            default:    larson_next = 5'b0_0010;
        endcase
    end


    // Sequential logic
    always @(posedge clk) begin
        if (scaler == 'd0) begin
            larson_reg <= larson_next;
        end
    end
*/

    reg  [5-1:0]                larson;    // = 5'b0_0001;

    // Sequential logic
    always @(posedge clk) begin
        if (BUTTON[0]) begin
            larson <= 5'b0_0001;
        end else if (scaler == 'd0) begin
            case (larson)
                5'b0_0001 : larson <= 5'b0_0010;
                5'b0_0010 : larson <= 5'b0_0100;
                5'b0_0100 : larson <= 5'b1_1000;
                5'b1_1000 : larson <= 5'b1_0100;
                5'b1_0100 : larson <= 5'b1_0010;
                5'b1_0010 : larson <= 5'b0_0001;
            endcase

        end
    end

    assign  LED = larson[3:0];

endmodule
```

Torstai 4.5.

- https://github.com/HelsinkiHacklab/digitaalitekniikka/blob/master/tutorials/1-combinatorial-logic/tutorial_top_1.v

https://github.com/JKN0/DAK-materiaalit-syksy-2015/blob/master/Labrat%C3%B6iden_toimintaselostukset.pdf

-----

```
// adder_top.v

// assign LED = { 1'b0, {1'b0,BUTTON[3:2]} + {1'b0,BUTTON[1:0]} };

`timescale 1ps / 1ps
`default_nettype none

module full_adder(
        input  wire                A,
        input  wire                B,
        input  wire                CIN,
        output wire                S,
        output wire                COUT
    );

    wire xor1_out;
    wire and1_out;
    wire and2_out;

    assign xor1_out = A ^ B;
    assign and1_out = xor1_out & CIN;
    assign and2_out = A & B;
    assign S = xor1_out ^ CIN;
    assign COUT = and1_out | and2_out;

endmodule


module tutorial_top(
        input  wire [4-1:0]        BUTTON,
        output wire [4-1:0]        LED
    );

    wire [1:0] inA;
    wire [1:0] inB;
    wire [2:0] sum;
    wire carry;

    assign inA = BUTTON[3:2];
    assign inB = BUTTON[1:0];

    assign LED[2:0] = sum;
    assign LED[3] = 1'b0;

    full_adder lsb(
        .A(inA[0]),
        .B(inB[0]),
        .CIN(1'b0),
        .S(sum[0]),
        .COUT(carry)
        );

    full_adder msb(
```

```
        .A(inA[1]),
        .B(inB[1]),
        .CIN(carry),
        .S(sum[1]),
        .COUT(sum[2])
    );

endmodule
```

- Pitayan GPIO liitin johon helppo laittaa johdot:
    - http://www.partco.fi/fi/liittimet/piikkirima-liittimet/harwin-liittimet/3250-molex-2x13.html
    - http://www.partco.fi/fi/liittimet/piikkirima-liittimet/harwin-liittimet/2750-molex-pin.html
- Magneettiliittimellä oleva micro USB johto
    - https://www.banggood.com/WSKEN-Magnetic-MINI1-2A-Micro-USB-Charging-Cable-for-Mobile-Phone-p-1110818.html?rmmds=myorder
- Tehtäväpohjat
    - https://github.com/HelsinkiHacklab/digitaalitekniikka/tree/master/tutorials/2-sequential-logic
    - jatkotehtävä:
        - muuta Larson skanneria siten, että päissä pysähdytään kaksi aikayksikköä

Torstai 27.4:

```
`timescale 1ps / 1ps
`default_nettype none

module top(
    input  wire [4-1:0]     BUTTON,
    output wire [4-1:0]     LED
  );

  assign LED = BUTTON + 'd1;

endmodule
```

```
`timescale 1ps / 1ps
`default_nettype none

module top(
    input  wire [4-1:0]     BUTTON,
    output reg  [4-1:0]     LED
  );

  always @* begin
    LED = 4'd0;

    if (BUTTON[0]) begin
```

```
        LED[0] = 1'b1;
      end
      if (BUTTON[0]) begin
        LED[1] = 1'b1;
      end
    end
endmodule
----
```

-

Torstai 20.4.:

JTAG ajurit PYNQille Ubuntuun:

- cd Xilinx/Vivado/2016.4/data/xicom/cable_drivers/lin64/install_script/install_drivers/ && sudo ./install_drivers
-

```
// digi_harjoitus_1_top.v

`timescale 1ps / 1ps
`default_nettype none

module digi_harjoitus_1_top(
      input  wire [4-1:0]        BUTTON,
      output wire [4-1:0]        LED
    );

    assign LED[0] = BUTTON[0];
    assign LED[1] = BUTTON[1];
    assign LED[2] = BUTTON[2];
    assign LED[3] = BUTTON[3];

endmodule
```

--Kurssin GitHub

- https://github.com/HelsinkiHacklab/digitaalitekniikka

--

PYNQ labin verkkoon Kiina-USB-WiFi -nappulalla (toistaiseksi toimivuus vähän satunnaista):

 # Kytketään WiFi USB-nappula ja katsotaan sen tunniste

xilinx@pynq:~$ **dmesg -wH**
[+0.054243] rtl8192cu 1-1:1.0 **wlx0013ef6000c7**: renamed from wlan0

 # Ota wlxXXXXXXX talteen (tässä esimerkissä oli wlx0013ef6000c7)

xilinx@pynq:~$ **sudo nano /etc/network/interfaces**

 # Lisää tiedoston loppuun, korvaten wlXXXXX kohdat mitä dmesg sattui sanomaan:
auto wlXXXXX
allow-hotplug wlXXXXX
iface wlXXXXX inet dhcp
  wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf

 # Tehdään uusi tiedosto jos ei löydy, sisällöksi jotain tämmöistä
xilinx@pynq:~$ **sudo nano /etc/wpa_supplicant/wpa_supplicant.conf**

```
network={
   ssid="Hacklab"
   psk="{tähän labin verkon salasana}"
   proto=WPA2
   key_mgmt=WPA-PSK
   pairwise=CCMP
   group=CCMP
   auth_alg=OPEN
}
```

 # Komennetaan uusi interface ylös
xilinx@pynq:~$ **sudo ifup -a**

Tänne pitäisi kohta ilmestyä IP-osoite
xilinx@pynq:~$ **ifconfig**

wlx0013ef6000c7 Link encap:Ethernet HWaddr 00:13:ef:60:00:c7
        inet addr:192.168.110.**XXX** Bcast:192.168.110.255 Mask:255.255.255.0
...

Torstai 6.4.:

--

// zturn_top.v

```
`timescale 1ps / 1ps
`default_nettype none

module zturn_top(
        input  wire [4-1:0]        SWITCH,
        input  wire [8-1:0]        PMOD_5,

        output wire                LED_R,
        output wire                LED_G,
        output wire                LED_B
    );

    assign LED_R = ~SWITCH[0];
    assign LED_G = ~SWITCH[1];
    assign LED_B = ~SWITCH[2];

endmodule
```

--

```
// pynq_top.v

`timescale 1ps / 1ps
`default_nettype none

module pynq_top(
        input  wire [2-1:0]        SWITCH,
        input  wire [4-1:0]        BUTTON,
        output wire [4-1:0]        LED,
        output wire [6-1:4]        LED_R,
        output wire [6-1:4]        LED_G,
        output wire [6-1:4]        LED_B
    );

    assign LED[0] = BUTTON[0];
    assign LED[1] = BUTTON[1];
    assign LED[2] = BUTTON[2];
    assign LED[3] = BUTTON[3];

    assign LED_R[4] = SWITCH[1];
    assign LED_G[4] = SWITCH[1];
    assign LED_B[4] = SWITCH[1];

    assign LED_R[5] = SWITCH[0];
    assign LED_G[5] = SWITCH[0];
    assign LED_B[5] = SWITCH[0];

endmodule
```

--

```
// pitaya_top.v
```

```
`timescale 1ps / 1ps
`default_nettype none

module pitaya_top(
    input  wire [16-1:0]    CON_E1,
    output wire [8-1:0]     LED
  );

  assign LED[0] = CON_E1[0];
  assign LED[1] = CON_E1[1];
  assign LED[2] = CON_E1[2];
  assign LED[3] = CON_E1[3];
  assign LED[4] = CON_E1[4];
  assign LED[5] = CON_E1[5];
  assign LED[6] = CON_E1[6];
  assign LED[7] = CON_E1[7];

endmodule
```

--

Board tiedostot Vivadoon: https://github.com/HelsinkiHacklab/digitaalitekniikka

--

 # PYNQ serial
Log to Pynq from Mac
screen -U /dev/tty.usbserial-00002014B 115200
(paina enter niin se tulostaa promptin)

Log to Pynq from Windows with Putty serial
Putty Connection type: valitse Serial, Serial line: COM11 (tai mikä sattuu olemaan oikea) ja aseta speed 115200
Paina Connect

#Pynq MAC osoitteen vaihto
ifconfig eth0 down
ifconfig eth0 hw ether 66:66:66:66:66:66
ifconfig eth0 up

Asetetaan teminaaliin värit ja koko
export TERM=xterm-256color && resize

# Sallitaan FPGA:n ohjelmointi kaikille
sudo chmod a+w /dev/xdevcfg

 # Z-turn serial:

Log to Z-turn from Ubuntu with serial

screen -U /dev/ttyUSB0 115200
(paina enter niin se tulostaa promptin)

Asetetaan teminaaliin värit ja koko
export TERM=xterm-256color && resize

# Asennetaan SSH-palvelin
sudo apt-get install openssh-server

Asetetaan root-käyttäjälle salasana
passwd

 # PYNQ SSH:

# macOS
sudo nano /etc/hosts
192.168.2.10		pynq

 # Z-turn SSH:

# macOS
sudo nano /etc/hosts
192.168.2.10		zturn

Torstai 30.3.:

Vaihda MAC-osoite:
ifconfig eth0 hw ether aa:bb:cc:dd:ee:ff

# Pynq mount fat
sudo mkdir /mnt/fat && sudo mount -o ro /dev/mmcblk0p1 /mnt/fat
sudo chmod a+w /dev/xdevcfg
cat /mnt/fat/design_1_wrapper.bit > /dev/xdevcfg

#Pynq mount USB

--

Esimerkiksi Ubuntusta PYNQiin
screen -U /dev/ttyUSB0 115200

Asetetaan teminaaliin värit ja koko
export TERM=xterm-256color && resize

alias rs='set noglob; eval `resize`'
rs # jatkossa kun ruudun kokoa on muutettu

```c
// yksinkertainen nappula -> ledi interaktio pynq-z1:lle
// (vaatii pynq-z1 defaultti overlayn ohjelmoiduksi, en muista mistä se löytyy tai mihin
// deviceen se tulee kirjoittaa.. mutta python-kalut tekevät sen)
// gcc ledbtn.c -o ledbtn && sudo ./ledbtn
// -- cut --
#include <stdio.h>
#include <sys/mman.h>
#include <stdint.h>
#include <unistd.h>

#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

#define MAP_SIZE 4096UL

int main(void)
{
	const int LEDS_OFFSET0 = 0x8;
	const int LEDS_OFFSET1 = 0xC;

	int memfd = open("/dev/mem", O_RDWR | O_SYNC);
	uint8_t* leds = (uint8_t *)mmap(0, MAP_SIZE, PROT_READ|PROT_WRITE, MAP_SHARED,
memfd, 0x41200000);
	uint8_t* btns = (uint8_t *)mmap(0, MAP_SIZE, PROT_READ|PROT_WRITE, MAP_SHARED,
memfd, 0x41210000);

	fprintf(stderr, "leds %p btns %p\n", leds, btns);

	leds[LEDS_OFFSET1] = 0;
	while(1)
	{
		leds[LEDS_OFFSET0] = btns[0];
	}
}
// -- cut --
```

```c
// gcc -std=gnu11 -Wall -Wextra -pedantic -O4 xadc.c -o
xadc && ./xadc
#define __USE_FILE_OFFSET64
#include <stdio.h>
#include <limits.h>
#include <stdint.h>
#include <assert.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/mman.h>
#include <fcntl.h>
#include <unistd.h>

volatile uint32_t * mmap_dev_mem(off_t offset, size_t length) {
```

```
   int fd = open("/dev/mem", O_RDWR);
      if (fd == -1) {
      perror("open");
      return NULL;
   }
   volatile uint32_t * ptr = (volatile uint32_t*)mmap(NULL, length, PROT_READ | PROT_WRITE,
MAP_SHARED, fd, offset);
      close(fd);
     if (ptr == MAP_FAILED) {
          perror("mmap");
          return NULL;
     }

   return ptr;
}

void munmap_dev_mem(volatile uint32_t * ptr, size_t length) {
   if (-1 == munmap((void*)ptr, length)) {
      perror("munmap");
   }
}


int main() {

      volatile uint32_t * xadc = mmap_dev_mem(0x43C00000, 64*1024);

      printf("ZYNQ XADC: TEMP       VCCINT      VCCAUX      VCCBRAM     VCCPINT
VCCPAUX     VCCDDRO     VP/VN       VREFP       VREFN\n");

          while(1) {
             printf("        %7.3f °C  %7.4f V   %7.4f V   %7.4f V   %7.4f V   %7.4f V   %7.4f V   %7.4f V
%7.4f V   %7.4f V   \n",
                (uint16_t)xadc[0x200/4]*0.00769192612943 - 273.15,     // TEMPERATURE
                  (uint16_t)xadc[0x204/4]*0.00004578754579,             // VCCINT
                (uint16_t)xadc[0x208/4]*0.00004578754579,             // VCCAUX
                (uint16_t)xadc[0x218/4]*0.00004578754579,             // VCCBRAM
                (uint16_t)xadc[0x234/4]*0.00004578754579,             // VCCPINT
                (uint16_t)xadc[0x238/4]*0.00004578754579,             // VCCPAUX
                (uint16_t)xadc[0x23c/4]*0.00004578754579,             // VCCDDRO
                (uint16_t)xadc[0x20c/4]*0.00004578754579,             // VP/VN
                (uint16_t)xadc[0x210/4]*0.00004578754579,             // VREFP
                 (int16_t)xadc[0x214/4]*0.00004578754579              // VREFN
                );
             sleep(1);
          }

      return 0;
}
```

Torstai 23.3.:

Board tiedostot Vivadoon: https://github.com/HelsinkiHacklab/digitaalitekniikka

Kortin luonti linuxilla:
fdiskillä uusi partitio ja laittaa type/id c (W95 FAT32 (LBA)
formatoi allaolevalla komennolla
mkfs.vfat -F32 /dev/OMAPARTITIO

-- hello.c

```
#include <stdio.h>
#include "platform.h"
#include "xil_printf.h"

volatile int kala = 0;

int main()
{
   init_platform();

   while (1) {
         print("Hello World - suovula\n\r");
         for (int i = 0; i < 10000000; i++) kala++;
   }

   cleanup_platform();
   return 0;
}
```

SSH ohjeet Pitayalle: https://wiki.redpitaya.com/index.php?title=SSH_connection

--

Tehtäviä:

mikä on 7-sarjalaisen Zynqin seuraavien tietotyyppien koko käyttäen sizeof()-operaattoria
- char
- int
- long
- float
- double
- uint8_t
- uint32_t
- int64_t
- size_t
- ssize_t

mitä vastaavasti omalla koneella kokeiltuna?
 - mitkä asiat vaikuttavat tuloksiin?

--

Muunna 0xBAADBEEF binääriluvuksi.

Vihje: tee HEX-BIN lunttitaulukko ensin paperilla, 16 riviä

Z-turn MYIR Board
usko
karpio
AnttiH
TorstiL
ara
Teppo2
Jaakko

guttula?

Konekanta:

suovula: macOS 16GiB RAM, 100GB SSD vapaana
latsku: Ubuntu: 16BG RAM, >100GB SSD vapaana
zokol: Kubuntu on VMWare: 8GB RAM, 300GB SSD vapaana
mangelis: Ubuntu 8GB RAM, SSD 22G avail
Yliolli: Windows 10 8GB RAM 100GB SSD
timo Windows 10 16GB RAM 615GB HDD vapaana
karpio win 10 4Gb 100 ssd
AnttiH: macOS 8 GB RAM, 200GB SSD vapaana
guttula: debian, 8GB RAM, 80GB vapaana
Kremmen: Wintoosa 10, 16GB, 120GB vapaana, Vivado 2016 asennettu jo aikaa sitten.
thjt: i7, Win10, 8GB, SSD ~200GB tilaa
ara: Win10, 6GB, ~80GB
Jari: Windows 10, 8 GB, n. 150GB SSD vapaana
Janne Windows 10 4GB 100 GB SSD
Josu: Windows 10, 4GB RAM,   vapaana 100GB (SSD)
Knorad: W10, i5, 16GB RAM, SSD 150GB vapaa
usko Windows 10, 4GB RAM 210GB vapaana
Teppo2 Debian 8 / 8Gt
Jaakko Desian 8 / 4Gt
EKH-ELL-i: ubuntu 3G RAM, SSD 24GB vapaana