

SoC Design Flow Tutorial

Barbara Gigerl, Rishub Nagpal

October 4th, 2023

Overview

- **Goal:** Run a bare-metal program on the CPU communicating with the HW on the FPGA



Overview

- **Goal:** Run a bare-metal program on the CPU communicating with the HW on the FPGA
- How can we do it?
 1. Prepare FPGA board (Zybo Board w/ Zynq SoC)
 2. Build FPGA HW
 3. Build SW for HW
 4. Combine everything in a boot image
 5. Copy the boot image to the FPGA
 6. Run the application



Important note



Things could go wrong - and will. Don't expect anything to work out-of-the-box.

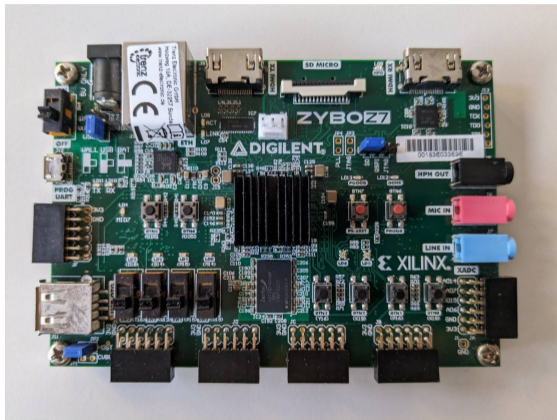
Install Vivado and Vitis (SDK)

- Xilinx Vivado: IDE for creating hardware designs
- Xilinx Vitis (SDK): SDK for embedded software
- Support for Xilinx FPGAs
- Installation: <https://www.xilinx.com/support/download.html>
 - Use version 2023.1
 - Use Linux Self Extracting Web Installer
 - Select Product to Install: first option (Vitis)
 - Save (a lot of) disk space: untick "Vitis Model Composer", devices for Alveo, devices for Kira SOMs, 7 Series, UltraScale, UltraScale+, Versal ACAP

Possible Installation Problems

- Make sure to install `libtinfo5` and beforehand!
- Useful log files: `~/Xilinx/xinstall/xinstall_*.log`
- See also:
<https://blog.lazy-evaluation.net/posts/linux/vivado-2018-3-buster.html>

Zybo Z7-20 FPGA Board



- Zynq 7020 SoC
- Processor: 650Mhz dual-core Cortex-A9
- Memory: 1 GB
- Peripherals:
 - Ethernet, USB, MicroSD slot
 - Video: VGA and HDMI
 - Audio: headphone out, microphone and line in
 - GPIO: 6 pushbuttons, 4 slide switches, 5 LEDs, 2 RGB LEDs

Configuration of Zybo Z7-20 FPGA Board

- Power source
 - JP6
 - Set to "USB"
- Boot source
 - JP5
 - Boot from SD card ("SD")

Prepare to start Vivado

- Source Vivado environment settings:
`source <VIVADO_ROOT>/Vivado/2023.1/settings64.sh`
- Install cable drivers:
 - `cd <VIVADO_ROOT>/Vivado/2023.1/data/xicom/cable_drivers/`
 - Execute install script.
 - Communication devices of board: `/dev/ttyUSB0`, `/dev/ttyUSB1`
 - Check if user has privileges to access devices (e.g. is member of dialout group)

Board files

```
ERROR: [Board 49-71] The board_part definition was not found
for digilentinc.com:zybo-z7-20:part0:1.0. The project's board_part
property was not set, but the project's part property was set to
xc7z020clg400-1. Valid board_part values can be retrieved with the
'get_board_parts' Tcl command. Check if board.repoPaths parameter
is set and the board_part is installed from the tcl app store.
```

- Clone <https://github.com/Digilent/vivado-boards>
- Copy vivado-boards/new/board_files including all subdirectories to <VIVADO_ROOT>/Vivado/2023.1/data/boards/
- Further details: https://digilent.com/reference/programmable-logic/guides/installing-vivado-and-sdk?redirect=1#installing_digilent_board_files

Building the FPGA Hardware



HW design in
VHDL/Verilog/...



Constraint file (.xcd)

Building the FPGA Hardware



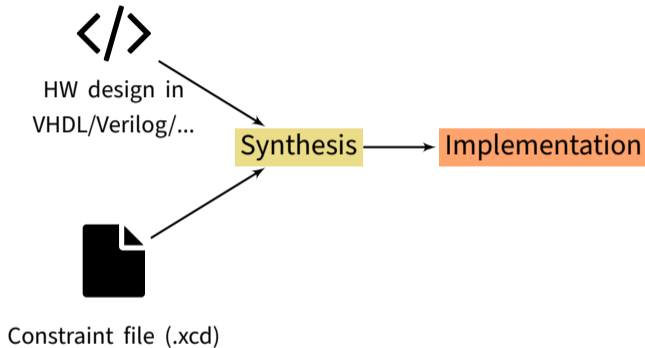
HW design in
VHDL/Verilog/...

Synthesis

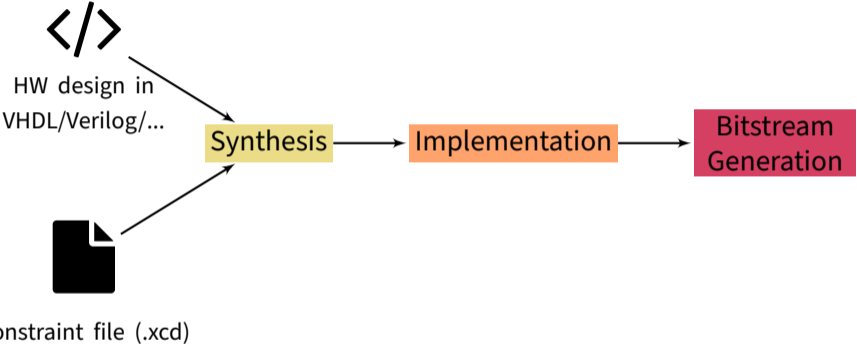


Constraint file (.xcd)

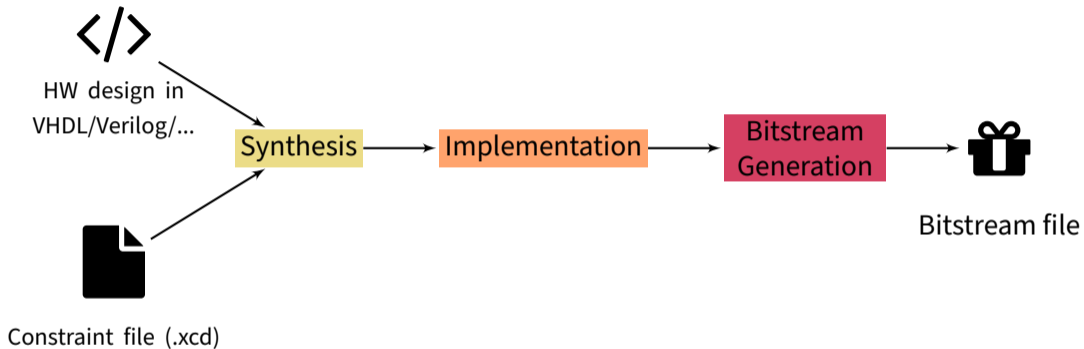
Building the FPGA Hardware



Building the FPGA Hardware



Building the FPGA Hardware



Synthesis

- Input 1: hardware design in any HDL (VHDL, Verilog, System Verilog, ...)

Synthesis

- Input 1: hardware design in any HDL (VHDL, Verilog, System Verilog, ...)
- Input 2: constraints file
 - Connects top level I/O ports to *real* hardware pins
 - Pins can be found in the specification
https://reference.digilentinc.com/_media/zybo:zybo_rm.pdf

Synthesis

- Input 1: hardware design in any HDL (VHDL, Verilog, System Verilog, ...)
- Input 2: constraints file
 - Connects top level I/O ports to *real* hardware pins
 - Pins can be found in the specification
https://reference.digilentinc.com/_media/zybo:zybo_rm.pdf
- Compile the HDL sources into an architecture-specific netlist

Synthesis

- Input 1: hardware design in any HDL (VHDL, Verilog, System Verilog, ...)
- Input 2: constraints file
 - Connects top level I/O ports to *real* hardware pins
 - Pins can be found in the specification
https://reference.digilentinc.com/_media/zybo:zybo_rm.pdf
- Compile the HDL sources into an architecture-specific netlist
- Check: code syntax, hierarchy, constraints (area, performance, power)

Synthesis

- Input 1: hardware design in any HDL (VHDL, Verilog, System Verilog, ...)
- Input 2: constraints file
 - Connects top level I/O ports to *real* hardware pins
 - Pins can be found in the specification
https://reference.digilentinc.com/_media/zybo:zybo_rm.pdf
- Compile the HDL sources into an architecture-specific netlist
- Check: code syntax, hierarchy, constraints (area, performance, power)
- Each gate is replaced by an architecture-specific block according to cell library:
 - Example: replace AND gate #1537 by macro LUT4, a 4-input Look-Up Table
 - Specification: https://www.xilinx.com/support/documentation/sw_manuals/xilinx2012_2/ug953-vivado-7series-libraries.pdf

Synthesis

- Input 1: hardware design in any HDL (VHDL, Verilog, System Verilog, ...)
- Input 2: constraints file
 - Connects top level I/O ports to *real* hardware pins
 - Pins can be found in the specification
https://reference.digilentinc.com/_media/zybo:zybo_rm.pdf
- Compile the HDL sources into an architecture-specific netlist
- Check: code syntax, hierarchy, constraints (area, performance, power)
- Each gate is replaced by an architecture-specific block according to cell library:
 - Example: replace AND gate #1537 by macro LUT4, a 4-input Look-Up Table
 - Specification: https://www.xilinx.com/support/documentation/sw_manuals/xilinx2012_2/ug953-vivado-7series-libraries.pdf
- Output: netlist tailored towards the FPGA on our ZYBO Board

Implementation

- Input: Netlist

Implementation

- Input: Netlist
- Floor planning, placement and routing, optimizations for specific architecture

Implementation

- Input: Netlist
- Floor planning, placement and routing, optimizations for specific architecture
- Example: Netlist element *AND #1537* which requires LUT4 is assigned *LUT4 #20A5* on the FPGA

Implementation

- Input: Netlist
- Floor planning, placement and routing, optimizations for specific architecture
- Example: Netlist element *AND #1537* which requires LUT4 is assigned *LUT4 #20A5* on the FPGA
- Output: map of netlist elements to FPGA blocks → FPGA configuration

Bitstream Generation

- Input: FPGA configuration

Bitstream Generation

- Input: FPGA configuration
- Transforms map into the bitstream (.BIT) format

Bitstream Generation

- Input: FPGA configuration
- Transforms map into the bitstream (.BIT) format
- Other formats: (.RBT, .BIN, .HEX, ...)

Bitstream Generation

- Input: FPGA configuration
- Transforms map into the bitstream (.BIT) format
- Other formats: (.RBT, .BIN, .HEX, ...)
- Programming FPGA = write bitstream file into FPGA memory

Bitstream Generation

- Input: FPGA configuration
- Transforms map into the bitstream (.BIT) format
- Other formats: (.RBT, .BIN, .HEX, ...)
- Programming FPGA = write bitstream file into FPGA memory
- On boot: FPGA loads configuration from memory and programs all LUTs

In Vivado...

1. Download base project:

https://extgit.iaik.tugraz.at/sip/zybo_z7_base_design

2. Generate Project: `cd <PROJECT_ROOT>/HW && vivado -source project.tcl`

3. Synthesis + Implementation + Bitstream Generation: **Flow - Generate Bitstream**
Note: takes around 5 minutes the first time

4. Export hardware to SDK project: **File - Export - Export Hardware**

- Include Bitstream

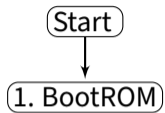
Start Vitis

- Simplest option: In Vivado, **Tools - Launch Vitis IDE**
- Keep default options for Exported Location and Workspace
- Otherwise, start Vitis similar to Vivado via console

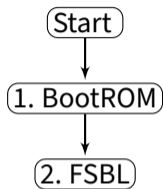
Boot Flow

Start

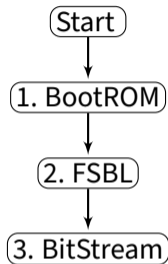
Boot Flow



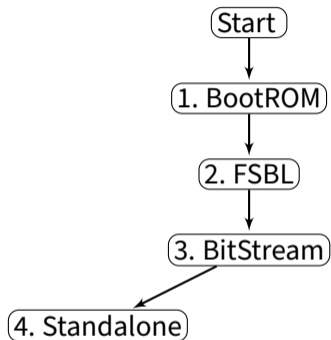
Boot Flow



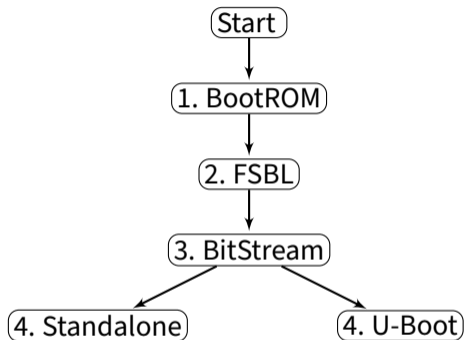
Boot Flow



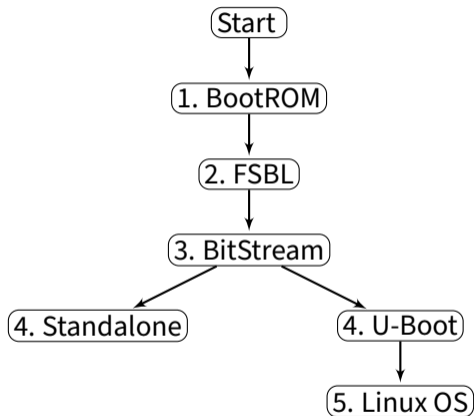
Boot Flow



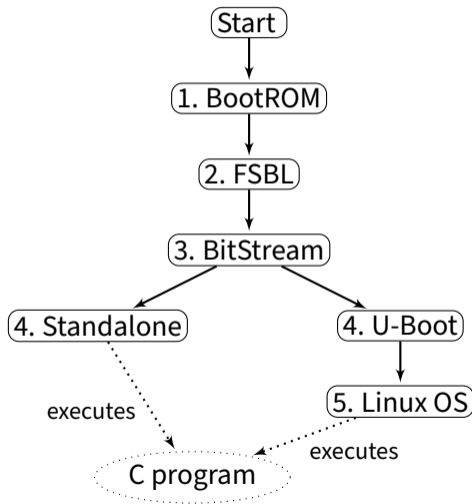
Boot Flow



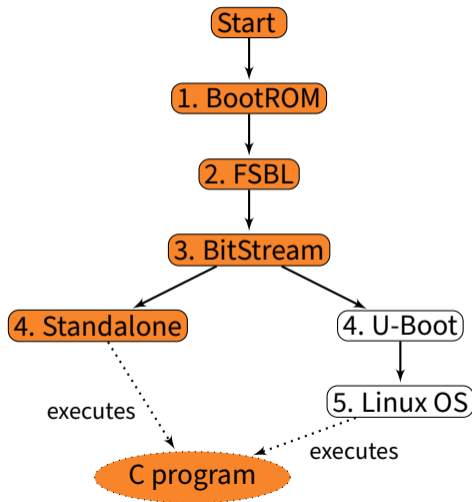
Boot Flow



Boot Flow



Boot Flow



FSBL and SW (in Vitis)

1. File - New - Application Project
 - Use default location
 - Platform: Create new platform from hardware...
 - Select Hardware Specification/XSA File from
`<PROJECT_ROOT>/HW/zybo_bsd/system_wrapper.xsa`
 - Generate boot components
2. Copy the base demo sources into the project's `src` directory
3. Build project

Creating a Boot Image (in SDK)

- Zynq SoCs combine FSBL, Bitstream and SW into one boot image (BOOT.bin)
- Select SW project, **Create Boot Image**
- Select correct paths for FSBL, Bitstream and SW (in this order!)
 - Should be correct by default
- Alternatively: run bootgen directly
`bootgen -image <IMAGE.BIF> -arch zynq -o <OUTPUT_FILE.BIN>`

Running the application via SD card

- Copy the boot image to SD card
 - First partition: FAT32, around 50 MB, used for boot files (BOOT.bin)
 - Second partition: ext4 or other, used as root file system and data storage
 - Use standard tools to create them (gparted, fdisk+mkfs ,...)
- Insert the SD card into the board
- Power up the board via SW4
- The red power LED (LD13) and the green FPGA programmed LED (LD12) should be on.
- Test image should be displayed via VGA/HDMI.

Communication with the board

- Use the virtual serial interface
- Typically: `/dev/ttyUSB1`
- Configuration: 115200 baud, 8 data bits, 1 stop bit, no parity
- `screen /dev/ttyUSB1 115200`

Running the application via JTAG

- Alternative to SD card
- Either correct jumper setting (JP5 to JTAG) or remove SD card
- Power up the board
- Program FPGA: In Vitis (SDK), `Xilinx - Program FPGA`
- Run binary: `Run - Run As - 1 Launch on Hardware (System Debugger)`
- Debug binary: `Run - Debug As - 1 Launch on Hardware (System Debugger)`
- Alternatively, via command line in xsdb console:
`cd <IMAGE_ROOT>/base_demo && xsdb run_base_demo.tcl`

Links and Resources

- Xilinx Installation Resources:

- <https://www.xilinx.com/support/download.html>

- <https://forums.xilinx.com/t5/Installation-and-Licensing/Installation-of-Vivado-2020-1-under-Centos-7-8-fails/td-p/1115482>

- https://www.xilinx.com/html_docs/xilinx2020_1/vitis_doc/aqm1532064088764.html

- ZYBO FPGA Board Reference Manual

- https://reference.digilentinc.com/_media/zybo:zybo_rm.pdf

- ZYBO-Z7 Reference Manual

- https://reference.digilentinc.com/_media/reference/programmable-logic/zybo-z7/zybo-z7_rm.pdf

- ZedBoard Reference Manual

- https://reference.digilentinc.com/_media/zedboard:zedboard_ug.pdf

