Comparing HLS with VHDL at the example of an FM Radio Receiver

Michael Wurm

MASTERARBEIT

eingereicht am
Fachhochschul-Masterstudiengang

Embedded Systems Design

in Hagenberg

im Juli 2021

Advisor:

DI (FH) Dr. Florian Eibensteiner

Declaration

I hereby declare and confirm that this thesis is entirely the result of my own original work. Where other sources of information have been used, they have been indicated as such and properly acknowledged. I further declare that this or similar work has not been submitted for credit elsewhere. This printed copy is identical to the submitted electronic version.

Hagenberg, July 15, 2021

Michael Wurm

Contents

Preface

Abstract

This should be a 1-page (maximum) summary of your work in English.

Kurzfassung

An dieser Stelle steht eine Zusammenfassung der Arbeit, Umfang max. 1 Seite. ...

Chapter 1

Introduction

1.1 Motivation

FM is a common RF signal that's available everywhere.
The demodulated signal is an audio signal, that can be listened to. This is subjectively more attractive than a generic data stream.

1.2 Broadcast Radio

Evolution from AM to FM because of certain advantages, etc.
Explain FM usage/existence nowadays (geographical; frequencies; devices; new standard DAB, etc)

1.3 Goal

put all knowledge that was accumulated during studies (HSD, ESD, mention subjects such as DSP) together in a project.
practical usage of DSP in an FPGA
develop an FM receiver and explain it, so that it can be followed in a tutorial with an affordable budget in hardware.
show multiple ways of how to implement the same thing, in different levels of abstraction. Weigh each ways' field of application, efficiency and applicability.

Thesis Design Decisions:

Main focus on system design.

How to achieve the same result in 3 (4?) different levels of abstraction?

1. GnuRadio (?)
2. Matlab/Python
3. HLS
   Implementation targeting Xilinx' ZedBoard.
   Optionally:

Intel as a comparison (synth only, no HW) - “How platform-independent is HLS?”

4. VHDL

Compare: - Implementation effort (difficulty) - Implementation time - Usefulness on Target Systems (uC, SoC, RasPi, etc) - others?

Chapter 2

Signal Processing Theory

2.1 Modulation from Baseband to RF

2.2 Modulation from RF to Baseband

2.3 Frequency Modulation (FM)

see literature/Present_lec6_AM_FM.pdf
see lectures out of HSD/ESD

2.3.1 Mathematical description

2.3.2 Frequency Band, Channel allocation/distribution

2.4 Algorithms for Digital FM Demodulation

see literature/FmDemodulator.pdf (Sect. 3.3) see literature/00476180 Digital FM Demodulator for FM, TV, and Wireless.pdf (Sect. II and III)

2.4.1 Baseband Delay Demodulator

2.4.2 Phase-Adapter Demodulator

2.4.3 Phase-Locked Loop (PLL)

2.4.4 Mixed Demodulator

Chapter 3

High-Level Synthesis

3.1 Introduction / State of the Art

3.2 Functionality (transform high-level code to HDL)

3.3 Language Support

3.3.1 C++

3.3.2 SystemC

3.4 Coding

3.4.1 Compiler Directives (#pragma's)

3.4.2 Data types

3.4.3 Functions

3.4.4 Loops

3.4.5 Conditional statements

3.5 Advantages / Disadvantages

easy and much faster to get complex HDL code, such as image/video processing, through libraries like OpenCV. However, it is less optimized, and a developer still needs to have a strong hardware background, to be able to understand what is being generated in hardware from x lines of code in HLS/Cpp.

Chapter 4

System Architecture/Concept

4.1 Block Diagram (with details)

4.1.1 describe blocks

4.2 Which parts in HLS and VHDL

4.3 Test Environment

Chapter 5

Implementation

5.1 Matlab/Python Model

in fixed point, close to hardware level algorithm

5.2 VHDL (no shortcut)

5.2.1 Channel Selection (IF to Channel-BB)

5.2.2 Phase Detector

5.2.3 other Elements

5.3 High-Level Synthesis

5.3.1 Channel Selection (IF to Channel-BB)

5.3.2 Phase Detector

5.3.3 other Elements

5.4 Common Testbench

5.4.1 Architecture (same tb for VHDL and HLS-generated HDL)

5.4.2 Framework cocotb, with ghdl compiler

Instantiate both HDL models in the testbench.
Display a direct comparison of outputs in graphs. This is practical, since the cocotb framework runs in python and graphs can be generated using Python's matplotlib.

5.5 Develop HLS compiler independent code (optional)

5.5.1 Challenges

Different compilers use different #pragmas, etc.
The #pragmas need to be within the code, at the appropriate positions, which makes it difficult to write compiler independent code....

Chapter 6

Test Results

6.1 Functionality

6.1.1 Implementation effort/time

6.1.2 Hardware Utilization

6.1.3 Others

Chapter 7

Deployment on Hardware

7.1 Hardware Platform

7.1.1 RTL2832u Dongle

7.1.2 ZedBoard

Appendix A

Technical Details

Appendix B

Supplementary Materials

List of supplementary data submitted to the degree-granting institution for archival storage (in ZIP format).

B.1 PDF Files

Path: /

thesis.pdf . . . . . Master/Bachelor thesis (complete document)

B.2 Media Files

Path: /media

*.ai, *.pdf . . . . . Adobe Illustrator files
*.jpg, *.png . . . . . raster images
*.mp3 . . . . . audio files
*.mp4 . . . . . video files

B.3 Online Sources (PDF Captures)

Path: /online-sources

Reliquienschrein-Wikipedia.pdf [1]

Appendix C

Questionnaire

Appendix D

LaTeX Source Code

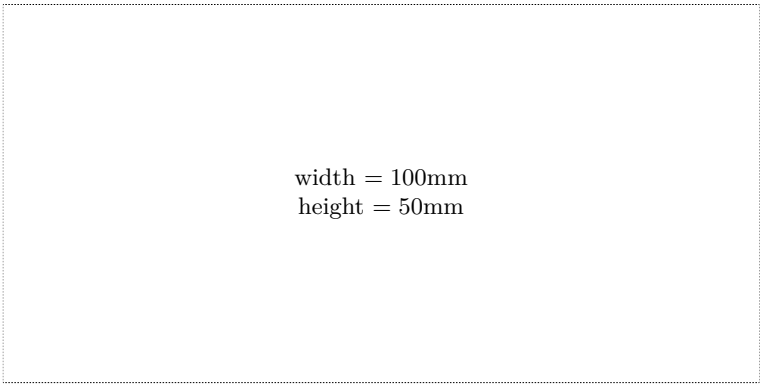
References

Online sources

[1] *Reliquienschrein*. Sept. 2018. URL: https://de.wikipedia.org/wiki/Reliquienschrein (visited on 02/28/2019).

Check Final Print Size

— Check final print size! —



— Remove this page after printing! —