**“High Level Synthesis of Digital Systems”**

**2023-2024**

**Prof. PERRI**

**Prof. FRUSTACI**

**FIR Filter Analysis**

|  |  |
| --- | --- |
| **Date** | <24/04/2024> |
| **Document** | Final Document |

|  |  |  |
| --- | --- | --- |
| **Full Name** | **ID** | **E-mail Address** |
| Giorgio Ubbriaco | 247284 | bbrgrg00h11d086x@studenti.unical.it |

Contents

[Definitions 4](#_Toc167134429)

[I. Tasks to be performed 5](#_Toc167134430)

[II. Abstract 6](#_Toc167134431)

[1. Introduction 7](#_Toc167134432)

# Definitions

# Tasks to be performed

# Abstract

# Introduction

# Solutions

Qui di seguito verranno illustrate e analizzate diverse implementazioni del filtro FIR. In particolare, verranno mostrati i report generati da HLS e Vivado così da effettuare ulteriori considerazioni a riguardo.

## Code Hoisting Solution

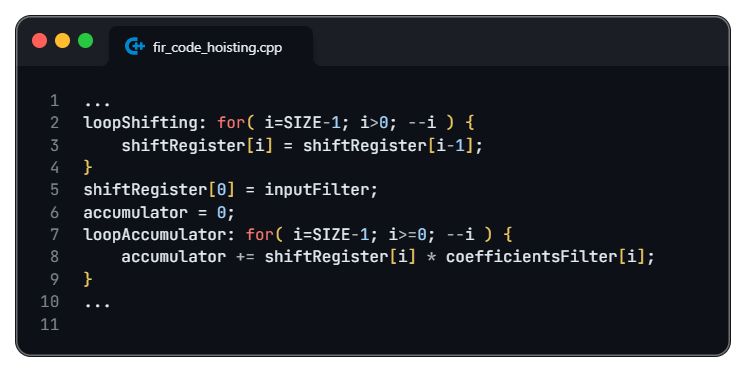
Dal momento che l’implementazione software del filtro FIR prevede un’istruzione condizionale, , dal punto di vista hardware questo corrisponderebbe ad un overhead a livello di risorse. Questo è dovuto al fatto che si dovrebbe leggere il valore dell’indice nella struttura dati che lo contiene e prevedere un comparatore così da confrontare tale valore con il valore e, pertanto, effettuare l’operazione logica di tra i bit che rappresentano la variabile e quelli rappresentanti il valore .



Quello che dovremmo aspettarci è che ci sia una riduzione dell’utilizzazione delle risorse prevista dal tool di HLS e dal tool di Vivado e, inoltre, una riduzione di un’unità del trip count rispetto alla soluzione non ottimizzata dovuto al fatto che il caso viene gestito al di fuori del ciclo for comportando, pertanto, un’iterazione in meno.

## Loop Fission Solution

La soluzione in questione prevede lo splitting del ciclo , previsto nell’implementazione software, in due parti. In particolare, l’operazione di shifting verrà effettuata in un ciclo denominato , mentre l’operazione di accumulo in un ciclo denominato . Questa divisione del ciclo in due cicli differenti permette al tool di HLS di effettuare ottimizzazioni indipendenti su entrambe le operazioni poiché presenti in un due cicli distinti. Bisogna notare però che, in questo caso, avrò due cicli e, pertanto, lo scheduling delle operazioni sarà differente dagli altri.



Quello che dovremmo aspettarci è che il trip count relativo al primo ciclo for sia pari a 10 poiché il caso viene gestito al di fuori del ciclo nella successiva istruzione similmente a come avviene nella soluzione del code hoisiting, mentre il trip count relativo al secondo ciclo ci aspettiamo sia pari a 11.

## Loop Unrolling Solution Factor=2 and Factor=4

### Manual Unrolling Solution

### Automatic Unrolling Solution

### Automatic Unrolling and Partitioning Solution

## Operation Chaining Solution

## Loop Pipelining Solution

## Bitwidth Optimization Solution

## AXI Solution