Report Lab Assignment
Running x264 on Host Processor
Extracting and Accelaring Kernel on Co-processor

Imara Speek (150000)
Aimee Ferouge (4014030)

2 november 2013

Date Performed: October 9-24, 2013
Instructor: Ir. Brandon

Samenvatting

10 lines of Abstract text

1 Introduction

For the lab a computational intensive kernel has to be extracted from a x264 software application, free software library for encoding video stream into H.236/MPEG-4 AVC format. The application is executed on a FPGA running a MicroBlaze host processor. By running the particular kernel on ρ-VEX co-processor, the execution time can be decreased by making use of hardware acceleration.

1.1 Getting Used to the Environment

The first lab is meant to get used to the project environment. A few input files are given to show the compile and run commands of the x264 application, by inserting a .y4m stream and creating a short .mkv movie. By adding the gprof flag to the compile command, a list is created of all functions ordered by their share of the total execution time (in percentage).

1.2 Detecting the Computationally Most Intensive Kernel

Profiling the x264 execution for the .y4m files that are provided by the lab, the following ranking is obtained: Given the chart in 1, we decide to extract the x264_pixel_satd_8x4 kernel, since its share in execution time increases as the files become larger. This kernel contains the Sum of Transformed Differences function for 8x4 pixel blocks.

1.3 Executing an Application on the Development Board and ρ-VEX

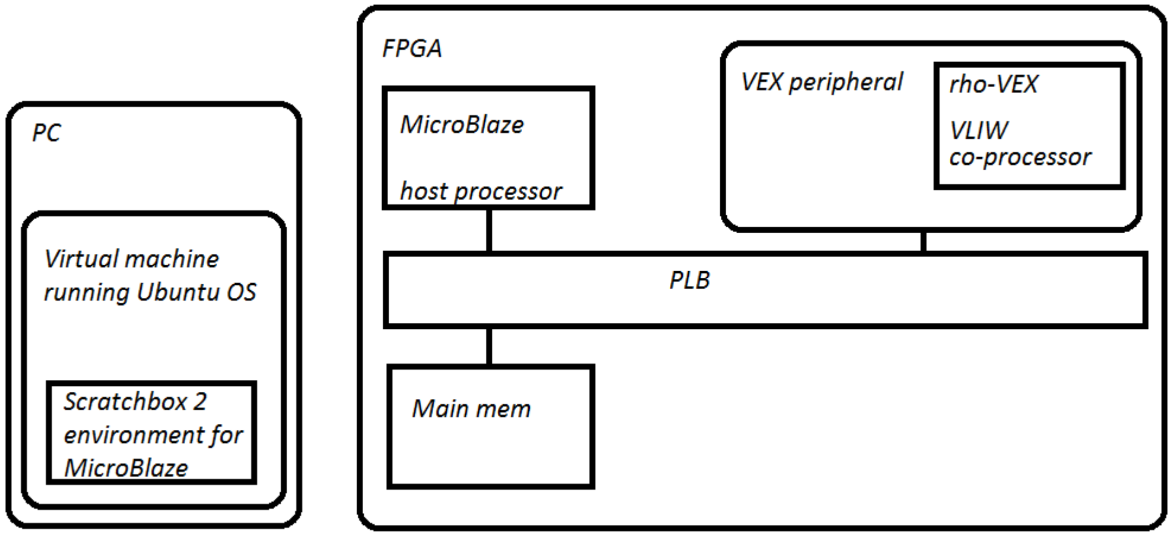
When executing ./configure, a file is created for configuration of the application. However, this config.mak file is made for applications running on the guest (Ubuntu). In order to configure for MicroBlaze, the config.mak file has to altered. All references to m32 have to be removed and the –DWORDS_BIGENDIAN flag has to be added to the CFLAGS variable. This has to be done everytime when configurating the application for MicroBlaze.

Input fie	Execution time (in sec)	Share (in %)	Kernel name
eledream_32x18_1.y4m	0.02	100.00 0.00 0.00	x264_analyse_init_costs x264_free x264_cabac_encode_desicion_c
eledream_64x32_3.y4m	0.03	66.67 33.33 0.00	x264_analyse_init_costs x264_pixel_satd_4x4 x264_pixel_satd_8x4
eledream_640x320_8.y4m	1.61	14.29 11.80 4.97	x264_pixel_satd_8x4 x264_get_ref x264_pixel_satd_x4_16x16
eledream_640x320_32.y4m	8.54	20.61 13.23 4.57	get_ref x264_pixel_satd_8x4 x264_pixel_satd_x4_8x8
eledream_640x320_128.y4m	29.86	17.48 14.17 6.56	get_ref x264_pixel_satd_8x4 x264_pixel_satd_x4_16x16

Tabel 1: Chart with computationally most intensive kernels for each input stream.

After doing this, the application now can be 'made' for MicroBlaze by first moving to the Scratchbox 2 environment for MicroBlaze and then execute the make command. Fig. **??** shows a block diagram of both the platforms.
En nu ga ik lunchen.

2 Rest


PC
Virtual machine running Ubuntu OS
Scratchbox 2 environment for MicroBlaze
FPGA
MicroBlaze host processor
VEX peripheral
rho-VEX VLIW co-processor
PLB
Main mem

Figuur 1: Block diagram of the Virtual Machine and the ERA platform