DYNAMIC ASSEMBLER FOR RECONFIGURABLE PROCESSOR

MOHAMAD AZMAN BIN BASIRAN

This Report is Submitted in Partial Fulfillment of Requirements for the Bachelor Degree of Electronic Engineering (Computer Engineering)

Faculty of Electronics & Computer Engineering
Universiti Teknikal Malaysia Melaka

JUNE 2016

UNIVERSTI TEKNIKAL MALAYSIA MELAKA
FAKULTI KEJURUTERAAN ELEKTRONIK DAN KEJURUTERAAN KOMPUTER

BORANG PENGESAHAN STATUS LAPORAN
PROJEK SARJANA MUDA II

Tajuk Projek : DYNAMIC ASSEMBLER FOR RECONFIGURABLE PROCESSOR

Sesi Pengajian :

1	5	/	1	6

Saya MOHAMAD AZMAN BIN BASIRAN mengaku membenarkan Laporan Projek Sarjana Muda ini disimpan di Perpustakaan dengan syarat-syarat kegunaan seperti berikut:

1. Laporan adalah hakmilik Universiti Teknikal Malaysia Melaka.
2. Perpustakaan dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan laporan ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. Sila tandakan (√) :

☐ **SULIT*** — *(Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)

☐ **TERHAD**** — **(Mengandungi maklumat terhad yang telah ditentukan oleh organisasi/badan di mana penyelidikan dijalankan)

☐ **TIDAK TERHAD**

Disahkan oleh:

________________________ (TANDATANGAN PENULIS)

________________________ (COP DAN TANDATANGAN PENYELIA)

Tarikh: 15 JUN 2016

Tarikh: 15 JUN 2016

“I hereby declare that the work in this project is my own except for summaries and quotations which have been duly acknowledge.”

Signature : ....

Author : MOHAMAD AZMAN BIN BASIRAN

Date : 15 JUNE 2016

"I acknowledge that I have read this report and in my opinion this report is sufficient in term of scope and quality for the award of Bachelor of Electronic Engineering (Computer Engineering) with Honours."

Signature	: ....
Supervisor's Name	: MR. SANI IRWAN BIN MD SALIM
Date	: 15 JUNE 2016

Dedicated to my beloved family especially father & mother, lecturer and to all my beloved friends.

ACKNOWLEDGEMENT

Thanks and praise to Allah, because of His grace, I have been able to finish this project successfully. The success and final outcome of this project required a lot of guidance and assistance from many people and I am extremely fortunate to have got this all along the completion of my FYP project. Whatever I have done is only due to such guidance and assistance and I would not to forget to thank them.

I respect and thank to my project guide Mr. Sani Irwan Bin Md Salim for giving me an opportunity to do the project work in FKEKK and took keen interest on my project and guide me all along, till the completion of my project by providing all the necessary information for developing a good assembler, and which made me complete the project on time. I am extremely grateful to him for providing such a nice support and guidance though he had busy with the phD's research.

I want to thank to my father, mother, and all my family for support me in surviving at UTeM and continuing my study till at this degree level. Thanks also goes to my friend who contributed ideas and teach a little bit of computer programming using the Visual Basic software.

ABSTRACT

The main objective for this project is to design a customizable assembler for dynamic instruction set architecture and to verify the assembler is compatible with the UTeMRISC processor architecture. In order to meet the objectives, a new assembler has been designed using Visual Basic 2015. The benchmark processor that is used for this project is UTeMRISC processor. In the development of processor architecture, one of the crucial part is the creation of a compatible assembler to the processor's instruction set architecture (ISA). Reconfigurable processor such as UTeMRISC03 requires a flexible assembler design in order to accommodate the modification being made to its ISA. The new assembler is capable in converting an assembly language program to its instruction word dictated by the processor's opcode file. The correct object files are also generated in line with the selected ISA width determined by the users. The object file is successfully loaded to the processor architecture in the FPGA platform in order to verify its compatibility. With the customizable feature achieved in this assembler design, the assembler would be beneficial as the main tool in the development of a complete package in a reconfigurable processor development in the future.

ABSTRAK

Objektif utama projek ini adalah untuk mereka bentuk pemasang disesuaikan untuk dinamik seni bina set arahan dan untuk mengesahkan pemasang itu serasi dengan seni bina pemproses UTeMRISC. Dalam usaha untuk memenuhi objektif, pemasangan baru telah direka dengan menggunakan Visual Basic 2015. Pemproses penanda aras yang digunakan untuk projek ini adalah pemproses UTeMRISC. Dalam pembangunan seni bina pemproses, salah satu bahagian yang penting ialah kejadian pemasang yang serasi untuk seni bina set arahan pemproses (ISA). pemproses konfigur seperti UTeMRISC03 memerlukan reka bentuk pemasang yang fleksibel untuk menampung pengubahsuaian sedang dilakukan untuk ISA itu. Pemasang baru mampu menukarkan program bahasa himpunan untuk perkataan arahannya ditentukan oleh fail Opcode pemproses. Fail objek yang betul juga dihasilkan selaras dengan lebar ISA yang telah dipilih oleh pengguna. Fail objek berjaya dimuatkan kepada seni bina pemproses dalam platform FPGA untuk mengesahkan keserasian. Dengan ciri yang disesuaikan dicapai dalam reka bentuk pemasang ini, penghimpun akan bermanfaat sebagai alat utama dalam pembangunan pakej yang lengkap dalam pembangunan pemproses pembentukan semula pada masa akan datang.

TABLE OF CONTENT

LIST OF TABLE

LIST OF FIGURE

LIST OF ABBREVIATIONS

FPGA	–	Field Programmable Gate Array
ROM	–	Read Only Memory
FKEKK	–	Fakulti Kejuruteraan Elektronik Dan Kejuruteraan Komputer
IEEE	–	Institute of Electrical and Electronics Engineers
ISA	–	Instruction Set Architecture
RISC	–	Reduce Instruction Set Computer
UTeM	–	Universiti Teknikal Malaysia Melaka
COE	–	Coefficient
LST	–	Listing
INT	–	Intermediate
FYP	–	Final Year Project
ERR	–	Error
CPU	–	Central Processing Unit
RAM	–	Random Access Memory
ALU	–	Arithmetic Logic Unit
IR	–	Instruction Register
PC	–	Program Counter
HDL	–	Hardware Description Language
HEX	–	Hexadecimal
ASM	–	Assembly
CD	–	Compact Disc
LSB	–	Lowest Significant Bit

LIST OF APPENDIX

CHAPTER I

INTRODUCTION

1.1 Project Overview

This project is to design a dynamic assembler for reconfigurable processor. Reconfigurable processor is a microprocessor with erasable hardware that can rewire itself dynamically. This allows the chip to adapt effectively to the programming tasks demanded by the particular software they are interfacing with at any given time. Within the internal architecture, the instruction set architecture act as the intermediate module between user program code (software) and hardware implementation. The processor's assembler must follow the changes of an Instruction Set Architecture (ISA). An assembler is a program that takes basic computer instructions and converts them into a pattern of bits that the computer's processor can use to perform its basic operations. The assembler program takes each program statements in the source program and generates a corresponding bit stream or pattern (a series of 0's and 1's of a given length). The assembler is design using Visual Basic and using two-pass assembly method. The output of the assembler as a coefficient file formatted will be used to burn into the FPGA implementation of the processor core.

1.2 Objectives

The objective of this project is to design and to develop a dynamic assembler for a reconfigurable processor.

1.3 Problem Statement

An assembler will not compatible with the processor if the reconfigurable processor’s architecture is change to another system configuration. An assembler of the processor is tied up to its architecture. The changes of new instruction set, bus modification, and memory expansion would change the system configuration inside the architecture of the processor. Processor could not run or there will have an error if architecture and assembler is not matched up. A customizable assembler could help to match up a multiple design of processor’s architecture. The machine code need to update with the latest version of the architecture. Reconfigurable processor required an assembler that compatible to the processor’s architecture. Thus, having a customizable assembler that producing an output of object files that matched up to the processor could solve this problem.

1.4 Scope

The assembler is developed using two-pass assembler approach. It could support 2 type of processor. The dynamic instruction set architecture is to be mode optional16-bit and 22-bit. Visual Basic 2013 used to design the assembler.

1.5 Report Structure

This thesis delivery the concept applied, method used, problem solving, finding analysis and result of Visual Basic. This thesis consists of five chapters and the following briefly describe what contents each chapter has.

In chapter 1, the purpose is to give reader an overall picture about what is actually this project doing. Introduction, objectives, scope of project, problem statement and summary of methodology are able to introduce this project to reader.

In chapter 2, the literature review of project is explained in detail. Study was done for existing assembler and the disadvantages of existing assembler were found. Describe about the introduction, problem statement, methodology used in the project, and the result that can be referred to improve in this project. Then, the theoretical concept that applied in the project based on the Visual Basic programming technique and target hardware also state in this chapter.

In chapter 3, the methodology of project is described. The processes of design are shown step by step. There are four main functions need to be designed which are load, tokenization, lexical analysis and decode to get the output. The assembler also will explain on how the main functions of existing assembler working. Then, the programming using the Visual Basic also will explain on how to design the new interface assembler.

In chapter 4, all results from project are included. The results are majority focus on assembler are using Visual Basic. That's mean the result assembler will show as Visual Basic interface. The COE file, LST file, and ERR file will generate after the assembler is done. The file also will test at FPGA processor to verify the file is creating well.

In chapter 5, a conclusion is made for the project that carried out in final year. The conclusion included project discovering, analysis achieved and future enhancement. Besides, the accuracy of project results will be concluded by comparing with objectives and problem statement. Finally, the important of this project to the target user will also describe.

CHAPTER II

LITERATURE REVIEW

2.1 An OpenCL Optimizing Compiler for Reconfigurable Processors [1]

2.1.1 Introduction

This project is about simple and efficient optimization techniques for an OpenCL compiler that targets reconfigurable processor. The target architecture consists of a general purpose processor core and an embedded reconfigurable accelerator with vector unit. The accelerator is able to switches its architecture between the VLIW mode and the Coarse Grained Reconfigurable Array (CGRA) mode to achieve high performance. One big problem of this architecture is programming difficulty and OpenCL can be a good solution. However, since OpenCL does not guarantee performance portability, hardware dependant optimization is still necessary. Hence, an OpenCL compiler framework that exploits the mode switching capability and vector units was developed. To measure the effectiveness of the techniques, the OpenCL framework was implemented and evaluates their performance with fourteen OpenCL benchmark applications.

2.1.2 Problem Statement

Conventional application processors in ASICs provide sufficient computational power to process various functions in current mobile devices. However, as the complexity of the applications for the mobile devices grows, the programmability of processors has become a major concern. Reconfigurable processors have been studied intensively as a programmable alternative to the ASICs.

2.1.3 Methodology

The reconfigurable processor typically consists of multiple basic components, e.g., functional units (FUs) and register files (RFs). Those components comprise processing elements (PEs), which are connected by dedicated wires for fast communication. The PEs and overall topology can be configured flexibly during runtime. In such architectures, the major challenge is programming difficulty. Application programmers need to explicitly control the accelerator, manage communications between the host and the accelerator, and guarantee coherence and consistency between memory blocks. Figure 2.1 shows the target architecture for the project. Figure 2.2 shows the compilation process and Figure 2.3 shows the vectorization example.

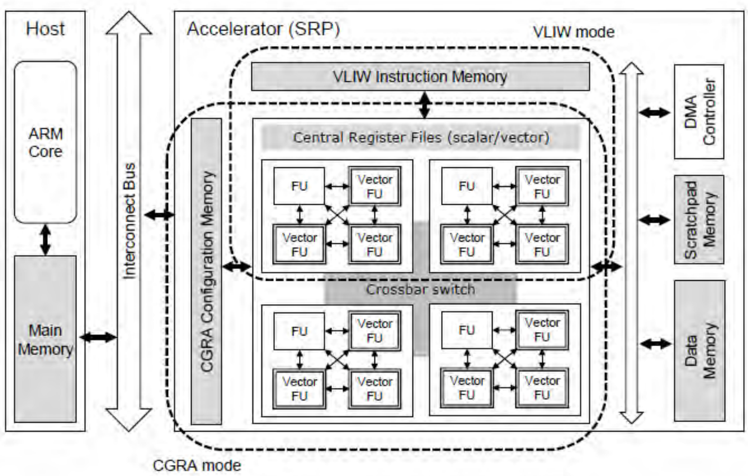

Figure 2.1: The target architecture

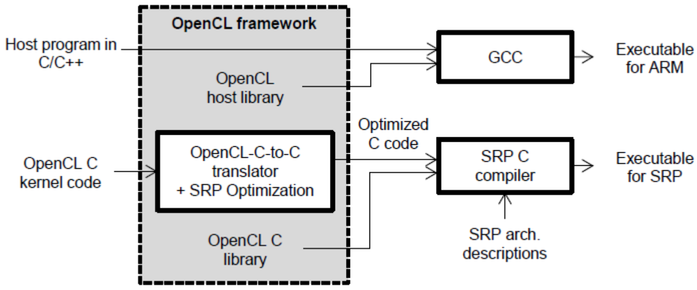

Figure 2.2: Compilation process

```
void vec_add(float *A, float *B, float P) {
  float4 *vA = (float4*)A;
  float4 *vB = (float4*)B;
  float4 vP = (float4)P;
  int4 vx, vy;
  for( int k = 0; k < local_size[2]; k++){
    for( int j = 0; j < local_size[1]; j++){
      #pragma rpcc cga
      for( int i = 0; i < local_size[0]; i += 4){
        vx = (int4)(goffset(0) + i) + (int4)(0, 1, 2, 3);
        vy = (int4)(goffset(1) + j);
        vB[vx[0]/4 + 64*vy[0] + 256] = vA[vx[0]/4] + vP;
      }
    }
  }
}
```

Figure 2.3: A vectorization example

2.1.4 Result

The baseline is C code obtained by base SnuCL translator, which is represented by the bar labelled Base. Most of the kernels are mapped to CGRA successfully and average speedup is 1.84. Six kernels of the benchmark applications are vectorised, average speedup of vectorization is 2.4 and the maximum speedup is 7.21. To increase efficiency of vectorization, the conditional statement optimization was applied, it is applied to Alpha Blending and it shows 4.21 speedup. The bar labelled WP+OPT shows the speedup over the baseline after applying work-item pruning and other optimizations. Average speedup of WP+OPT is 2.69 and the maximum speedup is 9.39. In case of Gaussian-Fan1 and Gaussian-Fan2, about half of the iteration range is pruned by work-item pruning. For *VectorAdd*, it can efficiently vectorize the code because control flow divergence is removed by work-item pruning. However, for the other kernels, performance improvements are limited to 1% - 3% because they are not vectorized and pruned iteration ranges are small. Figure 2.4 shows the output result of OpenCL kernel speedup.

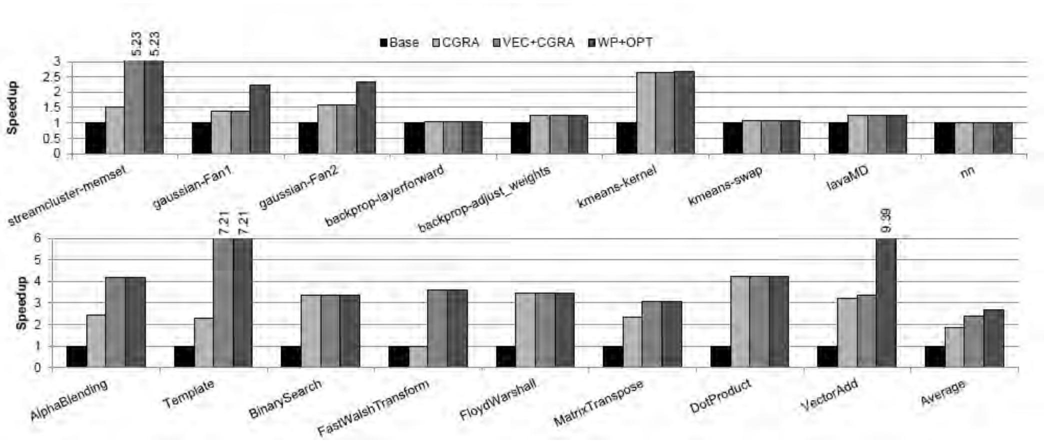

Figure 2.4: OpenCL kernel speedup

2.2 A Compiler Back-End for Reconfigurable, Mixed-ISA Processors with Clustered Register Files [2]

2.2.1 Introduction

Abstract—Reconfigurable tile-based architectures can dynamically interconnect several tiles in order to establish processor instances with varying resource, performance, and energy characteristics at run time. These flexible processor instances offer a new degree of freedom for adapting to changing applications' requirements while optimizing resource and energy consumption. The solution for dynamic interconnection of tiles requires a flexible Run-Time Scalable Issue-Width (RSIW) Instruction Set Architecture (ISA) that changes dependent on the configuration. In order to enable high-level programmability of the architecture in C/C++ a novel compiler back-end is needed.