Hardware and Software Partitioning using Genetic Algorithm in Image Processing Application

LOO FANG HEAN

UNIVERSITI SAINS MALAYSIA
2018

Hardware and Software Partitioning using Genetic Algorithm in Image Processing Application

by

LOO FANG HEAN

Thesis submitted in fulfillment of the requirements for the degree of Bachelor of Engineering (Electronic Engineering)

JUNE 2018

ACKNOWLEDGEMENTS

This dissertation is dedicated to everyone in the field of Genetic Algorithm in embedded system who embarks the journey of expanding the collection of knowledge and passion for Genetic Algorithm in embedded system.

Firstly, I would like to dedicate my appreciation and thank to my final year project and research supervisor, Assc. Prof. Dr Zaini Abdul Halim, for her patient guidance and advices throughout the entire period of my final year project. Her continuous support and precious advices are one of the factors in deciding the completion of my project and are highly appreciated. I feel proud and grateful to be one of the students who is under her supervision.

Next, I would like to thank the PhD student, Mr. Tan Earn Tzeh (one of the PhD student under the same supervisor as mine), for his precious advices and support for my final year project. He is always willing to share his precious experiences and knowledge to me.

Additionally, I want to thank all my course mates who study together with me throughout four years in my university life. They are always willing to spend their precious time with me, sitting together to exchange knowledge, discussing about the problems faced during the time of doing final year project and sometime brainstorming together to solve the problems.

Lastly, I would like to thank my family for giving me the loves and supports. Without their supports either financial or morale, I would not have successfully complete my final year project and thesis.

TABLE OF CONTENT

LIST OF TABLES

Page

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACO	Ant Colony Optimization
AI	Artificial Intelligence
ALU	Arithmetic Logic Unit
ARM	Advanced RISC Machine
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DAG	Directed Acyclic Graph
FPGA	Field Programmable Gate Array
GA	Genetic Algorithm
HDL	Hardware Description Language
HPS	Hard Processor System
HSCD	Hardware-Software Co-Design
HW	Hardware
ILP	Integer Linear Programming
MOPSO-CD	Multi-Objective Particle Swarm Optimization using Crowding Distance strategy
NSGA-II	Nondominated Sorting Genetic Algorithm
PE	Processing Engines
RAM	Random Access Memory
RGB	Red, Green, Blue
SA	Simulated Annealing
SoC	System-on-Chip
SOPC	System on a Programmable Chip

SVM	Support Vector Machine
SW	Software
WSGA	Weighted Sum Genetic Algorithm
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit

HARDWARE AND SOFTWARE PARTITIONING USING GENETIC ALGORITHM IN IMAGE PROCESSING APPLICATION

ABSTRACT

In this project, HW-SW Partitioning is used as a process to map each task of image processing application to be executed either in software (Hard Processor System, HPS) or hardware (Field Programmable Gate Array, FPGA). The framework for HW-SW Partitioning using Genetic Algorithm (GA) is developed in MATLAB. Total ten different combinations of GA parameters are used to test the developed framework. The GA parameters such as population size, crossover percentage and mutation percentage are varied to get the optimum combination of GA parameters. Three different HW/SW Partitioned Solutions are generated and the HW resources spent by first, second, and third solutions must not exceed the constraint value, Q = 341, Q = 681, and Q = 1022 respectively. The HW resource spent in HW/SW Partitioned Solution 1 (Q = 341) is 77.97% lesser than pure hardware solution. It is 4.65% faster than pure hardware solution and 26.6% faster than pure software solution. The HW resource spent in HW/SW Partitioned Solution 2 (Q = 681) is 50.29% lesser than pure hardware solution. It is 8.51% faster than pure hardware solution and 29.61% faster than pure software solution. The HW resource spent in HW/SW Partitioned Solution 3 (Q = 1022) is 45.01% lesser than pure hardware solution. It is 10.09% faster than pure hardware solution and 30.83% faster than pure software solution. Future work of this project is to implement the HW-SW partitioned solution in Altera DE1-SoC.

PEMBAHAGIAN PERISIAN PERKAKASAN DENGAN MENGGUNAKAN ALGORITMA GENETIK DALAM PEMPROSESAN IMEJ

ABSTRAK

Dalam projek ini, pembahagian perkakasan (HW)-perisian (SW) digunakan sebagai suatu proses yang menentukan setiap tugas aplikasi pemprosesan imej sama ada akan dalam perisian (HPS) atau dalam perkakasan (FPGA). Rangka kerja untuk Pembahagian HW-SW menggunakan Algoritma Genetik (GA) dibangunkan dalam MATLAB. Sejumlah sepuluh kombinasi parameter GA yang berlainan telah digunakan untuk menguji rangka kerja yang dibangunkan. Parameter GA seperti saiz populasi, peratusan crossover dan peratusan mutasi telah diubah untuk mendapatkan kombinasi parameter GA yang paling optimum. Tiga penyelesaian untuk pembahagian HW-SW yang berbeza telah dihasilkan dan sumber HW yang digunakan oleh penyelesaian pertama, kedua dan ketiga tidak boleh melebihi nilai kekangan, Q = 341, Q = 681, dan Q = 1022. Sumber HW yang diperlukan dalam penyelesaian pertama (Q = 341) adalah 77.97% lebih rendah daripada penyelesaian yang menggunakan HW sahaja. Penyelesaian ini adalah 4.65% lebih cepat daripada penyelesaian yang menggunakan HW sahaja dan 26.6% lebih cepat daripada penyelesaian yang menggunakan SW sahaja. Sumber HW yang diperlukan dalam penyelesaian kedua (Q = 681) adalah 50.29% lebih rendah daripada penyelesaian yang menggunakan HW sahaja. Penyelesaian ini adalah 8.51% lebih cepat daripada penyelesaian yang menggunakan HW sahaja dan 29.61% lebih cepat daripada penyelesaian yang menggunakan SW sahaja. Sumber HW yang dibelanjakan dalam penyelesaian ketiga (Q = 1022) adalah 45.01% lebih rendah daripada penyelesaian yang menggunakan HW sahaja. Penyelesaian ini adalah 10.09% lebih cepat daripada penyelesaian yang menggunakan HW sahaja dan 30.83% lebih cepat daripada penyelesaian yang menggunakan SW sahaja. Kerja masa depan projek ini adalah untuk melaksanakan penyelesaian HW-SW yang dipartisi dalam Altera DE1-SoC.

CHAPTER 1
INTRODUCTION

1.1 Research Background

Image processing is a method that analyzes digital data available inside an image in order to extract some useful information [1]. The main purpose of image processing is to distinguish the object in an image, seek for image of interest, create a better image, etc. Generally, image processing can be applied in many fields such as Automobile Industry, Electronic Industry, Medical, Robotics, etc.

Image processing can be a resource-consuming task. When the image that is undergoing processing is having high resolution, the process time and the resource consumption will be high. Therefore, this research is done to look for a solution that can provide low process time and at optimum level of resource.

Car plate identification is one of the applications in image processing. It may be implemented for security purposes, staff parking lot authentication, paying for toll, paying for parking fees, etc. With this system, a company can allow their staff to access the staff parking lot without using manpower to open the gate or allow the access to staff. This system will be more efficient than the ticket system that is currently used in a carpark of any mall. Besides, this system may replace the current toll system in Malaysia which is smart tag/ Touch N Go. With this system, the toll fees can be charged directly to car owner. The car owner who has cleared the toll debt is only allowed to change the road tax sticker.

Embedded System-On-Chip (SoC) incorporating Field Programmable Gate Array (FPGA) (Hardware) and Microprocessor/ Hard Processor System (HPS) (Software) are getting famous. Several SoC platforms with both Hardware and Software are being available commercially, which inclusive of Atmel's FPLSLIC family [2], Altera's low-cost CycloneTM II FPGA family [3], and Xilinx's Virtex II Pro [4].

A hybrid Microprocessor-FPGA embedded SoC platform has both processor and FPGA. However, most users are using either one of it for applications at the early stage of hybrid embedded SoC being available in market. FPGA provides parallel processing ability which has higher data processing rate than Microprocessor. However, FPGA consumes more resources to have similar performance as Microprocessor. This causes the application that uses FPGA only to have higher cost compared to the similar application implemented in Microprocessor [5-8]. Thus, using FPGA only for that application or using Microprocessor only for that application is not an effective solution. An optimal combination of hardware and software should be used in the application.

To have the combination, hardware-software partitioning technique needs to be implemented in the application. Hardware-software partitioning partitions several tasks to be implemented in hardware and implement the remaining tasks in software [6]. There are several algorithms used in Hardware-software partitioning such as Integer Linear Programming (ILP), Simulated Annealing (SA), Genetic Algorithm (GA) [9], etc.

GA is one of the methods available in Artificial Intelligence (AI). GA is used to solve both constrained and unconstrained optimization [10]. The process involved in GA is related to biological evolution where this algorithm will repeatedly modify the gene in the chromosome of an individual in the population. The gene mentioned can be represented in binary as a string of ‘0’s and ‘1’s [11].

In this project, Altera DE1-SoC platform as shown in Figure 1.1 will be used. This platform consists of FPGA and ARM Cortex A9 [3]. The image processing application is car plate identification. Instead of using only ARM Cortex A9 processor or only FPGA to perform the image processing, both ARM processor and FPGA will be used together to perform the image processing. The algorithm used to perform the hardware-software partitioning is Genetic Algorithm (GA). To use hardware-software partitioning, the task

required in the image processing is split into several smaller tasks that represented by genes which also known as nodes, performance of each node on FPGA and processor will be evaluated and analyzed based on the execution time and resource utilization.

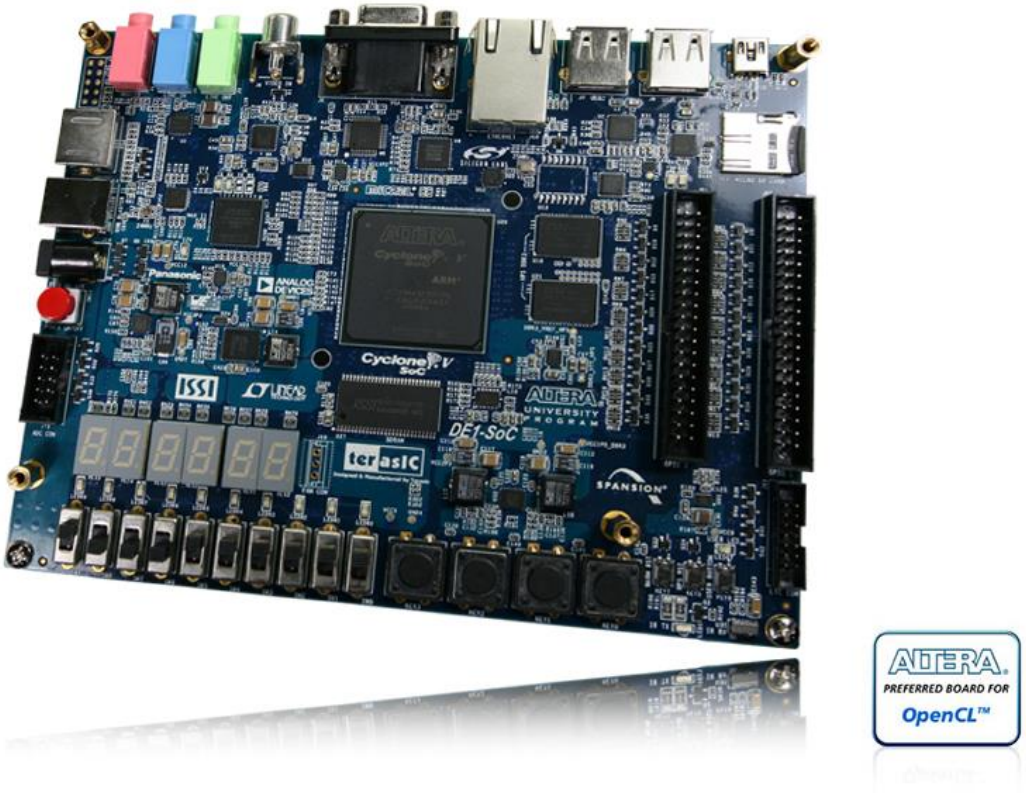


Figure 1.1: Altera DE1-SoC board [3]

1.2 Problem statement

Image processing is very famous in many applications such as face recognition, object detection, etc. However, these applications require a processing unit to process high resolution image to produce a better result. To perform image processing, majorities are using Central Processing Unit (CPU) to perform image processing on high resolution image where this method requires high processing time [12]. However, there are some people using FPGA to perform image processing. Although this method provides a shorter processing time, but the resource consumption is much higher [5-8]. Therefore, a better solution that capable to perform image processing faster and consume lower resources is needed to be explored.

Due to the mentioned reasons, an alternative approach to perform image processing based on Hardware-Software partitioning using GA will be investigated in this project. The image processing application is car plate identification.

1.3 Research objectives

The main goal of the project is to perform Hardware-Software Partitioning using GA algorithm for the image processing application. To achieve the goal, several specific objectives have to be fulfilled:

1. To develop a framework for Hardware-Software Partitioning using Genetic Algorithm for car plate identification using image processing method.
2. To investigate the performance of the Hardware-Software Partitioning in terms of processing speed and resource consumption for car plate identification.

1.4 Scope of project

In this research/project, the main objective is implementing Genetic Algorithm to perform the Hardware-Software Partitioning but the C code for HPS and the Verilog code for FPGA are not the main focus. GA is used to identify the best combination of the nodes to be executed in hybrid HPS-FPGA platform such as Altera DE1-SoC. This means that GA will decide which node to be executed in FPGA and which node to be executed in ARM processor (HPS). The nodes mentioned are referred as the tasks involved in the image processing application which is car plate identification. To achieve the objective, image processing tasks/nodes have to be identified and verified through Microsoft Visual Studio 2017 with the help of OpenCV 3.4.0.

The code used for image processing application in Microsoft Visual Studio is converted into Verilog code and C code. The Verilog code and C code will be implemented in FPGA and ARM processor of Altera DE1-SoC respectively. Data such as processing time and resources used for each task/node in FPGA are collected from Quartus Prime 17.1 and ModelSim 16.0 respectively. Data such as processing time used for each task/node in ARM processor is collected from the terminal in the Linux Ubuntu Desktop with Linux Kernel 4.5.

With the help of MATLAB, the framework for HW-SW Partitioning using GA is developed and only three GA parameters are varied which are population size, crossover percentage, and mutation percentage. The selection method is fixed at “Roulette-Wheel” method. The number of iterations and mutation rate are fixed at 100 and 0.02 respectively. The tabulated data will be processed by the developed Genetic Algorithm to create the best HW-SW Partitioned solution. Three different constraints are applied to GA which are Q = 341, Q = 681, and Q = 1022 where Q is the maximum hardware resource constraint. In other words, three different HW-SW Partitioned solutions will be generated

by GA and the hardware resources spent by first, second and third solutions cannot exceed Q = 341, Q = 681, and Q = 1022 respectively.

1.5 Thesis Outline

Overall, this report consists of five main chapters that describe the full details from introduction to conclusion of this project. Chapter 1 is the introduction which briefly describes the project background, problem statement, objectives and scope.

Chapter 2 is about the literature review of the past works done related to this project. Summary of the past studies on hardware performance for image processing, software performance for image processing, different algorithms for Hardware Software partitioning and different image processing techniques for car plate detection are presented in this chapter.

In Chapter 3, methodology and the project implementation will be discussed in detail. Flowcharts, image processing techniques and car plate identification algorithms used are elaborated. The implementation of the project will also be discussed in this chapter. List of devices and system operation are explained in detail.

Chapter 4 presents the result and discussion for this project. The performance of the image processing nodes in ARM processor and FPGA in terms of execution time and resource utilization will be presented. The performance of Hardware-Software Partitioned solution is compared Hardware-Software Non-Partitioned solution such as pure hardware implementation solution and pure software implementation solution. The result is analyzed and discussed.

Finally, chapter 5 presents the conclusion of this project. Summary on the project implementation and future works of this project are included.

CHAPTER 2
LITERATURE REVIEW

2.1 Overview

Recently, researchers have started to research on Hardware Implementation or Software Implementation or Hybrid Hardware-Software Implementation for image processing application in portable embedded platform. In this chapter, some fundamental knowledge relevant to this project has been discussed. Besides, previous work done on the image processing algorithm used in identifying car plate, algorithms to perform hardware/software partitioning and implementation of image processing on embedded platform like FPGA or microprocessor have been reviewed in this chapter as well.

2.2 Processor (CPU)

Processor which is also known as Central Processing Unit (CPU) is an electronic circuitry that executes the instructions of a program in sequential flow by performing arithmetic, logic and control operations. These are done by arithmetic logic unit (ALU), register and control unit. ALU in CPU performs arithmetic and logic operation, register in CPU stores the results generated by ALU, and control unit of CPU controls the flow of fetching and execution of instruction [13].

2.3 Field Programmable Gate Array (FPGA)

FPGA allows user to design the circuit. User can configure or design the circuit using Verilog code or VHSIC Hardware Description Language (VHDL) where VHSIC is known as Very High Speed Integrated Circuit. FPGA contains an array of programmable logic blocks and reconfigurable interconnects that allow the logic blocks to be wired together and Figure 2.1 shows the FPGA fabric structure that consists of logic blocks, I/O blocks and programmable interconnect.

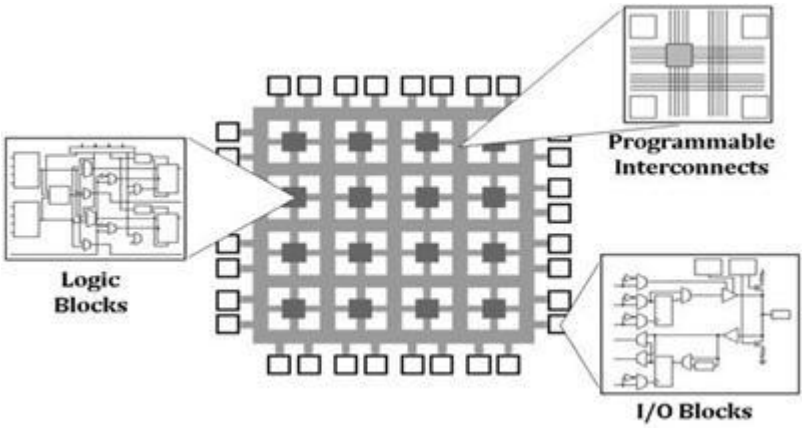


Figure 2.1: FPGA fabric structure [14]

FPGA can be configured into something as simple as an AND gate or it can be configured into something as complex as a multi-core processor. FPGA store the configuration in Random Access Memory (RAM) but not flash memory, therefore, the configuration is lost when FPGA lose the power supply. FPGA can achieve parallel processing because it allows user to configure circuit at gate level [15-16].

2.4 CPU-FPGA Hybrid Platform

Altera DE1-SoC is one of the CPU-FPGA hybrid embedded platforms that consists of both ARM-based Hard Processor System (HPS) and FPGA together with a high-bandwidth interconnect between HPS and FPGA. The HPS consists of 800MHz Dual-core ARM Cortex-A9 embedded core, peripherals and memory interfaces. This board consists of Ethernet networking, video capability that can display images through VGA port and DDR3 memory [17]. With this hybrid platform, user can implement design in HPS only by using software or implement in FPGA by using Hardware Description Language (HDL) or implement in both HPS and FPGA by using hardware-software co-design method.

2.5 Hard Processor VS Soft Processor

Hard processor is a type of processor system where all components have been fixed and the circuit cannot be reconfigured after it is manufactured while soft processor refers

to a type of processor system that is created using the available resources in FPGA where this system can be customized and reconfigured again anytime. Although soft processor has better flexibility than hard processor in terms of reconfigurable, but soft processor has higher power consumption and requires more resources than hard processor [5-8, 18].

2.6 Hardware Software Partitioning

Hardware-Software Partitioning is one of the most crucial design steps in Hardware-Software Co-Design (HSCD). There are several algorithms used in Hardware-software partitioning such as Integer Linear Programming (ILP), Simulated Annealing (SA), Genetic Algorithm (GA) [9], etc. These algorithms are used to decide which components of a system to be realized in hardware and which ones in software [6].

Hardware-software partitioning provides an optimal trade-off between time consumed and resources spent. It solves the problem arising when a system is being implemented in hardware only which is higher resources consumption or the problem arise when a system is being implemented in software only which is longer time consumed [5-7].

2.7 Genetic Algorithm (GA)

Today, GA is involved in solving several daily life problems. Many sales-based company has adopted a solution that uses GA as their most effective solution to the traffic and shipment routing problem [11]. GA is being used in robotic system [11], Encryption and Code Breaking [19], Finance and Investment Strategies [19], Optimizing Chemical Kinetic Analysis [19], Gene Expression Profiling [19], etc. To partition tasks into hardware and software to get short processing time and resource consumption within maximum resource available, GA algorithm is needed for hardware-software partitioning.

GA, a heuristic search algorithm that is inspired by Charles Darwin's theory on the natural evolution which is "survival of the fittest". This algorithm reflects the process of natural selection where the fittest individuals in a population are selected for reproduction in order to produce better offspring in next generation. This algorithm exploits historical information to direct the search into the region of better performance within the search space [20-21].

An individual is characterized by a set of parameters/variables/nodes which is known as gene. Genes are joined into a string to form a chromosome/solution of a system. Population is formed when a group of chromosomes exist [20]. Figure 2.2 illustrates the definition of population, chromosomes and genes used in Genetic Algorithm. In this project, the nodes are assigned to each task in the image processing task such as converting color image to grayscale image, converting grayscale image to binary image, etc. The entire application consists of many genes and is represented as chromosome.

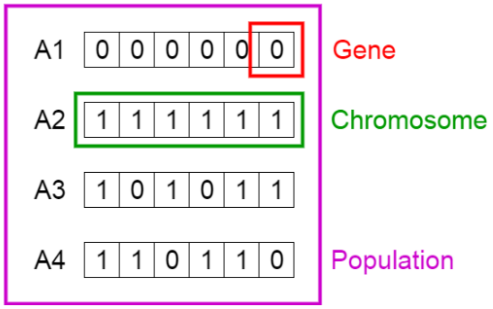
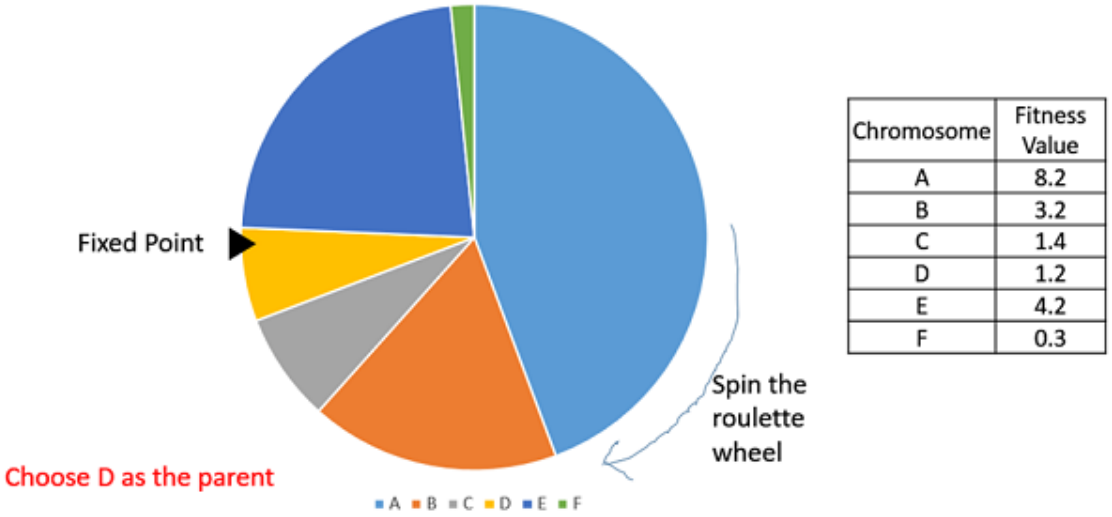


Figure 2.2: Population, Chromosome and Genes [20]

Genetic algorithm generally involves mutation, cross-over, fitness evaluation and selection [22]. “Selection process” is a process that selects candidates in the current population for mating process. “Cross-over process” is a process that breaks a pair of candidates’ into parts and remixed into two new candidates. Mutation is a process where a chromosome is randomly selected and the gene is altered from ‘0’ to ‘1’ or vice versa. Fitness evaluation is process where it evaluates the fitness of candidates in a population/ in that generation. The higher the fitness (low execution time), the better the candidate is.

There are three common types of “selection process” methods which are inclusive of Roulette Wheel Selection, Tournament Selection and random selection. In selecting the parent for cross-over using Roulette Wheel Selection, the circular wheel is divided according to the fitness value. Based on Figure 2.3, the candidate with higher fitness value will get higher portion in the circular wheel. The region which comes in front of the fixed point is chosen as parent. The same process is repeated for second parent.



Chromosome	Fitness Value
A	8.2
B	3.2
C	1.4
D	1.2
E	4.2
F	0.3

Figure 2.3: Visual Representation of Roulette Wheel Selection in GA [22]

In selecting the parent for cross-over using Tournament Selection, K number of candidates/chromosomes are randomly picked out of the population and the fittest candidate among K candidates is selected as parent. Based on Figure 2.4, the candidate with higher fitness value out of the three potential candidates is selected as parent.

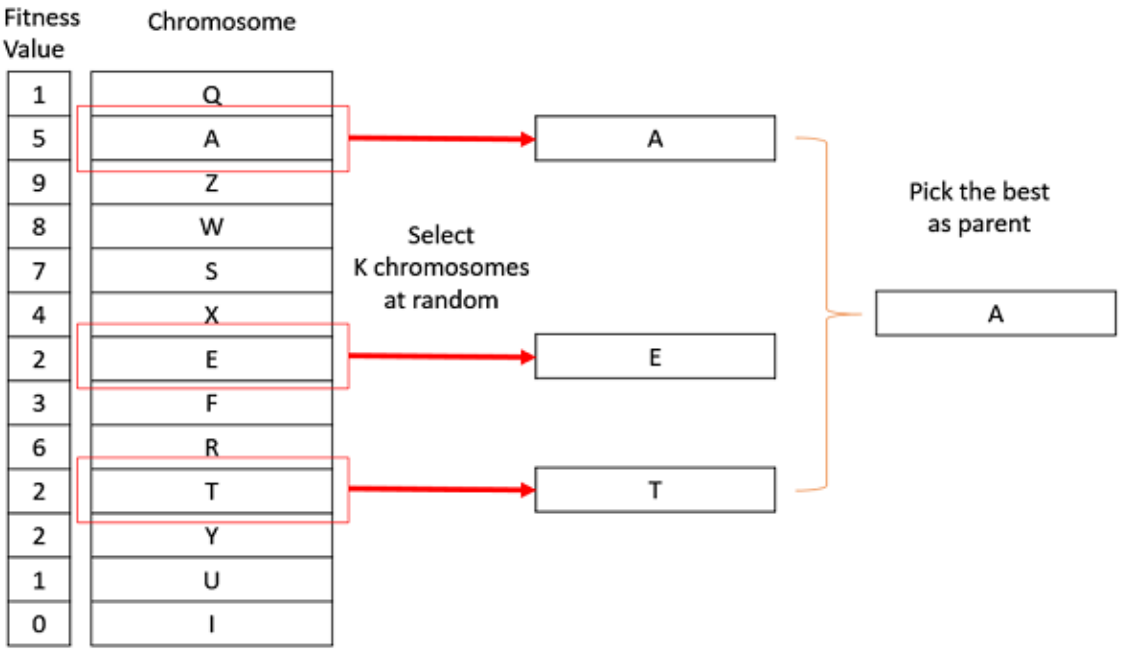


Figure 2.4: Visual Representation of Tournament Selection in GA [22]

In selecting the parent for cross-over using Random Selection, two random parents are selected for cross-over.

For cross-over process, there are three commonly used cross-over method. The first method is known as one-point crossover. A random crossover point is selected. The tails of two parents are swapped to get new off-springs as Figure 2.5 shown.

Figure 2.5: Visual representation of One-point crossover in GA [22]

The second method is known as Multi-Point Crossover. This method is a generalization of one-point crossover method wherein alternating segments are swapped as Figure 2.6 shown.

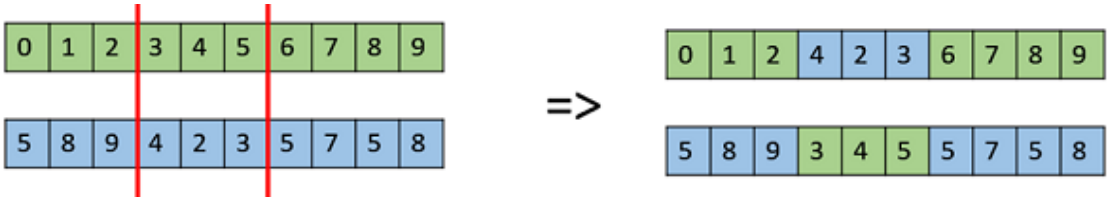


Figure 2.6: Visual representation of multi-point crossover in GA [22]

The third method is uniform crossover where each gene has potential to swap between parents. It can be said that coin is being flipped for each gene to decide whether or not it will be swapped between parents as Figure 2.7 shown.

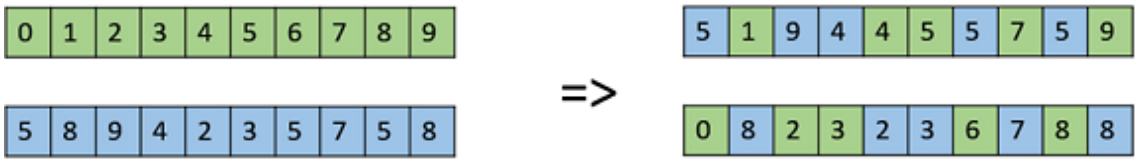


Figure 2.7: Visual representation of uniform crossover in GA [22]

The most common mutation method is bit flip mutation [22]. A random candidate is selected from the population. The selected candidate will have one or more random bit being flip. Figure 2.8 shows one candidate having one-bit flip mutation.

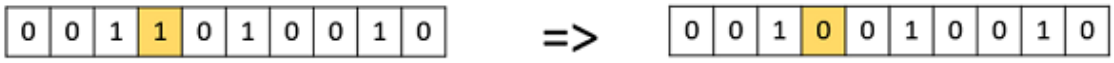


Figure 2.8: Visual representation of uniform crossover in GA [22]

In this project, after injecting the data collected such as time and resource consumed by each node on both FPGA and HPS, the Genetic Algorithm generates the fittest individual/chromosome for the hardware software partitioning. For example, the fittest individual/chromosome is "011". Assuming "1" is the node to run in FPGA/Hardware and "0" is the node to run in HPS/Software, then the hardware/software partitioning is having the image processing task to be performed in "HPS-FPGA-FPGA" if the fittest individual/chromosome generated is "011".

2.8 Image Processing

Image processing is a process to perform some operations on an image in order to extract some useful information from the image such as identify the number of pets in an image, etc. Image processing is also commonly used to enhance an image. It is a type of signal processing where the input of the system is an image and the output maybe an image. For example, an image can be undergoing image processing to find the contours of the objects in the image and the system outputs an image with the contours being marked [23].

An image is made up of many pixels. For example, a 640*480 (width*height or column*row) image is having 640 columns and 480 rows of pixel which made up of total 307200 pixels [24]. Each pixel in a color image contains three values which are Red, Green, Blue (RGB). These values range from 0 to 255 representing the intensity of the color at that pixel. A colored image can be imagined as a 3-layered 3-Dimensional Array where the row and column of each layer of the 3-Dimensional array are the height and width of an image respectively and the height/layer of the 3 layered 3-Dimensional Array are representing the RGB value each pixel carries [25].

A colored image can be converted into a grayscale image. The fundamentals of this conversion is converting the three values which are RGB value of each pixel into a single value. This conversion can be said as converting a 3-Dimensional array to a 2-Dimensional array. Therefore, each pixel only carries one grayscale value from 0 to 255 only. There are three common methods existed to convert a colored image to a grayscale image. The first method is lightness method where grayscale value in each pixel is calculated as Equation (2.1) [26].

$$\text{Grayscale} = \frac{\text{Max}(R,G,B)+\text{Min}(R,G,B)}{2} \tag{2.1}$$

The second method is average method where grayscale value in each pixel is calculated as Equation (2.2) [26].

$$\text{Grayscale} = \frac{R+G+B}{3} \tag{2.2}$$

The third method is luminosity method where grayscale value in each pixel is calculated as Equation (2.3) [26]. The output images of three methods are shown in Figure 2.9.

$$\text{Grayscale} = 0.21\,R + 0.72\,G + 0.07\,B \tag{2.3}$$

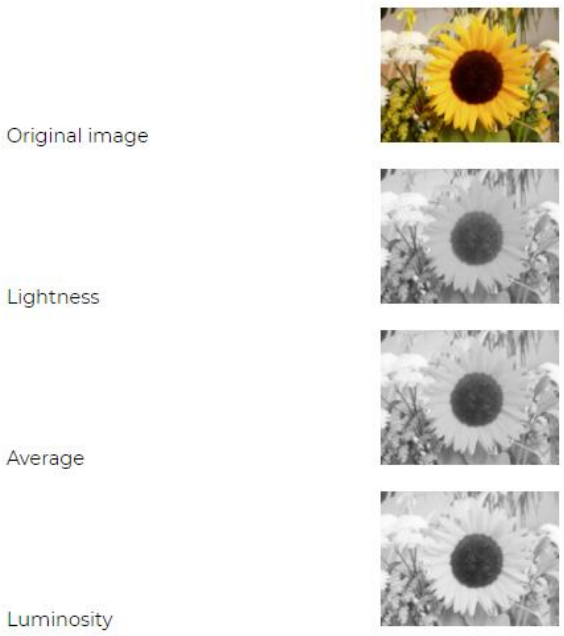


Figure 2.9: Converting colored image into grayscale image using three methods [26]

2.9 Related Works

Yoshihiro Shima [27] proposed a method to extract car plate region out of an image captured from the rear end at various car distances. The proposed system had extracted the plate candidate region by using dilation morphology and scoring based on both height-width ratio and number of connected components in the region. The proposed system classified the candidate region into two classes which were number plate and outlier. It used pre-trained 23-layers of Convolutional Neural Network (CNN) as image feature extractor and two Support Vector Machine (SVM) as classifier. Yoshihiro Shima proposed a method to extract the car plate region which is a combinational of both morphological image processing and deep learning. For the morphological image processing, the flow of the system is as shown in Figure 2.10.

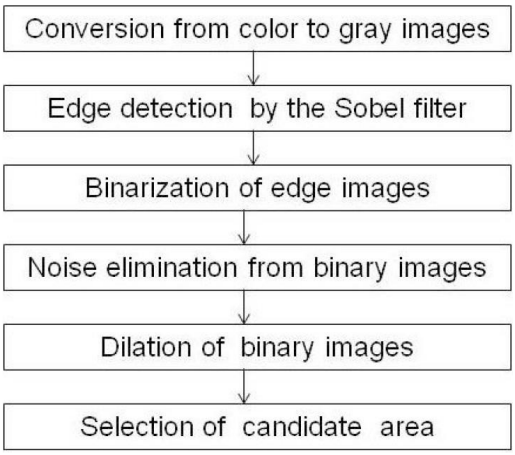


Figure 2.10: Flow of morphological image processing proposed by Yoshihiro Shima [27]

For the deep learning process, Yoshihiro Shima [27] had injected 269 samples of positive number plate images for pre-trained CNN and 370 negative training outlier images for SVM. In the first layer of the deep learning network, the first layer learned filters for capturing blob and edge features. For validating the results/performances, Yoshihiro Shima had tested to extract the car plate using morphological image processing with a success rate of 75% for 56 samples of image which is equivalent to 42 sample

images succeeded. Therefore, the researcher implemented combinational of both morphological image processing and deep learning in this research. With this combination, the success rate is increased to 89.7% for 126 samples of image where the algorithm had correctly identified the car plate for 113 sample images.

Anumol Sasi et al. [28] had researched an intelligent approach that is able to detect vehicular number plates automatically using three algorithms in plate localization for identifying the edges, in character segmentation and extraction algorithm and a hierarchical combined classification method. The algorithm is named as Ant Colony Optimization (ACO).

ACO algorithm has several limitations when it is used in edge detection such as random initial ant position in image. Anumol Sasi et al. [28] had modified the ACO such as assigning well-defined initial ant position. The character segmentation and extraction algorithm which used the concept of Kohonen Neural Network to identify the position and dimensions of characters was presented.

For the hierarchical combined classification method, the combination of both inductive learning and SVM for character recognition was proposed. However, in this paper, the image processing application is to extract the car plate from an image. Therefore, the flow of plate extraction/localization is important to be reviewed. For the plate localization, Anumol Sasi et al. [28] had proposed a flow as shown in Figure 2.11.

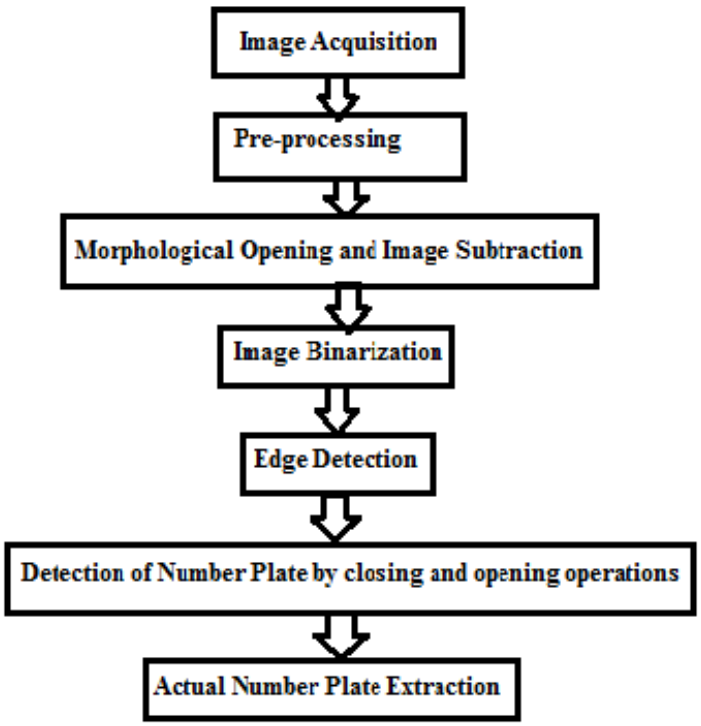


Figure 2.11:Flow of Plate localization proposed by Anumol Sasi et al. [28]

In the pre-processing stage, the image had been processed to improve the contrast and reduce the noise in the image. During this stage, the image had been converted from RGB image to grayscale image, noise reduction by median filtering and Adaptive Histogram Equalization. In the Morphological Opening operation, the adaptive contrast enhanced image had been processed to remove disc shaped structural element.

The morphological opened image was then subtracted from the adaptive contrast enhanced grayscale image to highlight the number plate region. The subtracted image had been converted into binary image using threshold method. The edge detection method used modified ACO algorithm. After the edge detection, the next step includes closing operation which is obtained by performing dilation first and then followed by erosion. Then the car plate had been extracted from the image after getting the coordinates of the plate.

Sajed Davoodnia and Mohammad Ghasemzadeh [29] both researched and proposed a new algorithm for locating the car plate by separating the blue channel from

a colored image and differentiating the blue channel from the grayscale image and threshold the image to generate a binary image. Based on Figure 2.12, the researchers separate the blue channel was because of the highlighted region in Figure 2.12. Candidate area was determined using geometrical properties such as area and length to width ratio. The researchers had done analysis using MATLAB environment to test the algorithm with 150 images of standard size of 640*480 pixels and different ambient light conditions and direct imaging of Iranian vehicles with national license plates. The researchers obtained 96.66% of efficiency and accuracy for the proposed algorithm.

Figure 2.12: An example of Iran's national license plates with different backgrounds [29]

However, based on Figure 2.13, this algorithm has imperfection which is having false detection on car or background with blue color and similar geometry property.

Figure 2.13: Disorder in plate positioning due to similarity with the body color of the vehicle [29]

Rajesh Kannan Megalingam et al. [30] researched and proposed a technique using image processing only for automatic license plate recognition. The researchers used a digital camera to capture color image with pre-defined resolution of 640*480 pixels. The plate region is extracted by the concept of connected components in the image (mathematical morphology). After extracting the plate region, the researchers proposed that the extracted plate region to undergo character segmentation and followed by character recognition. For the plate extraction, the flow is as shown in Figure 2.14.

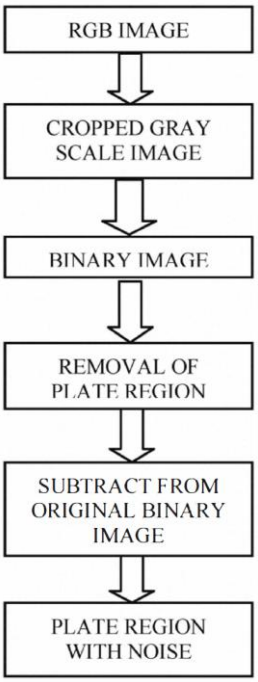


Figure 2.14: Flow of the plate extraction proposed by Rajesh Kannan Megalingam et al. [30]

Based on Figure 2.14, Rajesh Kannan Megalingam et al. [30] proposed the conversion of RGB image into grayscale image. Then the researchers proposed cropping the 640*480 pixels of grayscale image to remove unwanted boundary regions by removing first and last 60 rows from the image and first and last 180 columns from the image leaving the image to have the resolution of 280*360 pixels. Then the cropped grayscale image is converted into binary image by using threshold method. Thus, the plate characters are displayed in white pixels. Then the proposed technique looked for the

connected white components of less than or equal to the alpha-numeric character size used in the license plate and remove them from the binary image.

After removing the character, the image generated was subtracted from the binary image. This produced an image with those components which are less than or equal to the characters used in the license plate. Then the generated image had been processed to reduce noise by removing very small connected white pixels from the image. Then the researchers performed horizontal and vertical profiling (horizontal and vertical histogram) on the image to locate the coordinate of the plate. After locating the plate coordinate, the plate was extracted from the grayscale image.

Shuang Dou et al. [7] had proposed a model of embedded system that is extended such that resource contentions are taken into consideration. GA-based algorithm is proposed on the basic of the model. From the experimental results, the GA-based algorithm can be conveniently implemented for Hardware-Software Partitioning with resource contentions.

In their research, the results obtained in GA-based algorithm are compared with exhaustive search method. The application is modelled by considering a Data Flow Graph where nodes are derived from the functional specification internally represented as control/data flow graph, which can run either on hardware or on software. The execution time is divided into three parts which are execution time in hardware, execution time in software and time interfere with resource contention.

In their research, the Hardware-Software partitioning problem is modeled as nonlinear minimization problem. GA algorithm in their research has provided a better result as the system area is optimized with lower run time as compared to Exhaustive Search algorithm.

M.C. Bhuvaneswari and M. Jagadeeswari [9] had researched multi-objective optimization of minimizing two objectives area and the execution time of the partition. Performance metrics are calculated to validate and verify the efficiency of the algorithms. In the research, the embedded system to be partitioned is modeled as Directed Acyclic Graph (DAG). A DAG is a graph G (V, E), where V is the set of functional nodes/tasks and the edge, E, is the data dependency between the two tasks with $V \in \{v_1, v_2 \ldots v_i\}$ and $E \in \{e_1, e_2 \ldots e_i\}$.

Each task is associated with five integer values: (1) HW-execution time, t_H, which is the time required to execute the task on the HW module; (2) HW implementation of the task required area, C_H, on the HW module; (3) SW-execution time, t_S, which is the time required to execute the task on the processor; (4) the SW implementation of the task required memory, C_S, on the software module; and (5) the communication costs, C_C, which refers to the delay in transfer of data between HW and SW tasks.

The researchers illustrated three multi-objective optimization algorithms: Weighted Sum Genetic Algorithm (WSGA), Nondominated Sorting Genetic Algorithm (NSGA-II), and Multi-Objective Particle Swarm Optimization using Crowding Distance strategy (MOPSO-CD) which are applied for solving HW/SW partitioning problem. The result of this research indicated that NSGA-II is effective in solving Hardware-Software partitioning problem and provided good results for most benchmark circuits. Figure 2.15 is a sample population for Hardware Software partitioning problem.

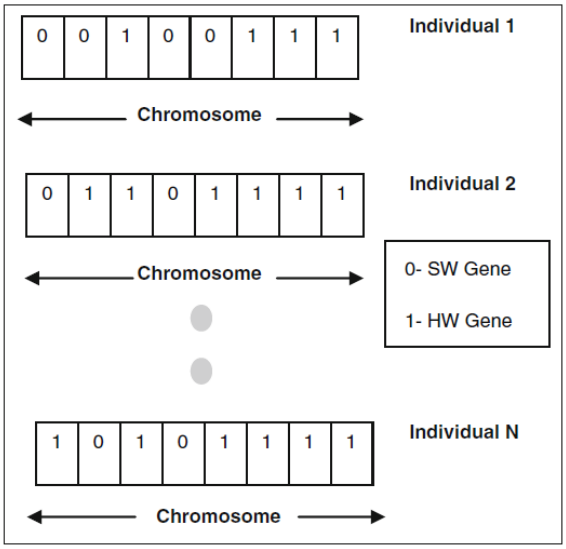


Figure 2.15: Sample population mentioned by M.C. Bhuvaneswari and M. Jagadeeswari for Hardware Software partitioning problem [9]

Li Luo et al. [31] had researched GA for hardware-software partitioning on a heterogeneous multicore System on Chip which has several different types of Processing Engines (PE). GA is used for four task graphs to simulate the hardware/software partitioning. Task graph is also known as Directed Acyclic Graph (DAG) with weight. 5 definitions are declared so that task graph can be used to describe the system. The GA algorithm is implemented as shown in Figure 2.16.