Particle Swarm Optimization Based Placement and Routing of Hardware Tasks in 2D Homogeneous FPGAs

Ms.B.Premalatha, Ms.D.Divya, Ms.N.Abinaiya, Ms.S.Monisha

Abstract— Ever increasing demand of flexibility, speed and low power consumption for complex applications such as embedded systems, image processing, video processing, cryptography, etc. initiate the runtime reconfigurable devices as an interesting one in the field of advanced computing systems. Reconfigurable devices such as Field Programmable Gate Arrays (FPGAs) consist of limited resources but high performance in their computing and also exhibit high parallelism. Hence, the hardware resources of high density FPGAs should be properly managed and allocated. Various algorithms are developed in various research works by concentrating in reducing task rejection rate and to have less fragmentation. This paper deals with using an heuristic approach for placement and routing of hardware tasks for partially reconfigurable FPGAs and observe the performance.

Index Terms—2D homogeneous FPGAs, High performance computing, Optimization algorithms, Partially reconfigurable FPGAs, Placement, Resource management, Routing

1. INTRODUCTION

Most commercial FPGA architectures have the same basic structure, a two-dimensional array of programmable logic blocks that can implement a variety of bit-wise logic functions, surrounded by channels of wire segments to interconnect logic block I/O. The internal architecture as well as interconnections of FPGA can be reconfigured to match the needs of a given application.

Partial reconfiguration is the selective updating of a sub-array of an FPGA's programmable logic and routing resources while the remainders of the device’s programmable resources continue to function without interruption. Thus, an FPGA does not have to be halted in order to have its function partially reconfigured. The main advantage of partial reconfiguration is that it offers the fastest way to change an active FPGA circuit, since only those parts that need to be reconfigured are interrupted.

- *B.Premalatha is currently working as an Assistant Professor in Departmrnt of Electronics and Communication Engineering, Sri Krishna College of Tecnology,Coimbatore-42, TamilNadu, India,PH-+91-9994714595.E-mail: premdeepa@rediffmail.com.*
- *D.Divya, N.Abinaiya, S.Monisha, are currently pursuing Bachelor’s degree program in Electronics and Communication Engineering, Sri Krishna College of Tecnology,Coimbatore-42, TamilNadu,India,PH-+91-9944625670.*

E-mail:divyadevendrann@gmail.com.

The most important synthesis steps in FPGA are Scheduling, Placement and Routing.

1.1 Scheduling

Real-time scheduling, i.e., determining the sequence of execution of tasks with deadlines, is a major problem in critical embedded systems. Their design must ensure that the timing constraints imposed by the surrounding physical system can be guaranteed. The (inherently complex) worst-case response time and feasibility analysis of tasks under scheduling algorithms like earliest deadline first is hence of great importance.

1.2 Placement

The FPGA’s placement is to create a placed configuration of logic blocks that can be interconnected successfully in a subsequent routing step within the available routing resources.

1.3 Routing

The routing phase interconnects specified sets of terminals, i.e., the signal nets of the design, by wiring within routing regions that lie between or over the functional units. (A signal net is a set of the module output terminals and the corresponding module input terminals which need to be connected to each other using routing).

2. PARTICLE SWARM OPTIMIZATION (PSO)

Placement is an NP-complete problem. NP class of problems consists of the all the decision problems whose positive solutions can be verified using a polynomial time on a non-deterministic machine. NP-complete problem means there is no known polynomial time algorithm exists. Approximate solutions for NP-complete problem can be found using heuristic methods.

The optimization problem in hand is attempted using Particle swarm optimization, which is one of the very recent tools introduced by Kennedy (a social psychologist) and Eberhart (an electrical engineer) [3]. The initial concept of the swarm intelligence imitates the swarm of birds. It tries to locate the optima based upon the social interaction as well as the individual cognition.

In PSO, a particle is defined to be all any possible solution of the problem in a solution space and we use a group of particles at different locations to find the optimum solution. The movement of the particle is decided by two things: Individual cognition and social interaction.

1. Each particle keeps track of its coordinates in the solution space which are associated with the best solution (fitness) that has achieved so far by that particle. This value is called personal best , ***pbest***.
2. Another best value that is tracked by the PSO is the best value obtained so far by any particle in the neighborhood of that particle. This value is called ***gbest***.

Both the above two factors and the present velocity of the particle affects the velocity in the next iteration. The velocity is added to the present location of the particle to get the next location which will help it move towards the best location(gbest), achieved by the swarm, while still looking for an even better location(improving pbest).
The basic concept of PSO technique lies in accelerating each particle towards its pbest and the gbest locations at each time step [1],[6]. The velocity and position of the particles are changed according to the equations (1) and (2) respectively.

$$v_{i+1} = v_i + c_1 * rand_1(\ ) * (pbest_i - x_i) + c_2 * rand_2(\ ) * (gbest - x_i) \quad (1)$$

$$x_{i+1} = x_i + v_i \quad (2)$$

where $\boldsymbol{rand}(\ )$ is any random in the range of (0,1) generated each time when the function is evaluated.
$\boldsymbol{c_1}$ and $\boldsymbol{c_2}$ are two constants known as acceleration constants
Velocity is kept in a range of **[**$\boldsymbol{-v_{max}}$, $\boldsymbol{v_{max}}$**]**

In a modified form of PSO, Shi and Eberhart [3] introduced inertia weights to the velocity equation in order to control the scope of search in an efficient way and to reduce vmax. In this new form the modified velocity equation is given by,

$$v_{i+1} = w_i * v_i + c_1 * rand_1(\ ) * (pbest_i - x_i) + c_2 * rand_2(\ ) * (gbest - x_i) \quad (3)$$

Where $\boldsymbol{w_i}$ is the inertia weight which can be changed in each iteration to control the scope of the search. When $\boldsymbol{w_i}$>1 the search step is higher and it can go inspect the behavior of different locations in the search. When $\boldsymbol{w_i}$<1, it performs a rather intensive search operation with a small step size. So it is better to keep the value of $\boldsymbol{w_i}$ initially higher and then gradually reducing it to a small value to perform a rigorous search. We have used the variation of the weight governed by the following equation,

$$w = (\max\ it - iter) / \max\ it \quad (4)$$

where $\boldsymbol{iter}$ is the current iteration number and $\boldsymbol{max\ it}$ is the total number of iterations to be done.

3. PLACEMENT BASED ON PSO

In this paper, the following assumptions are made for the preliminary PSO based placement: (a). the interconnection distances between the Configurable Logic blocks (CLBs) in FPGA are considered in terms of normalized units and (b). the connections from I/O ports are neglected.

For placement based on PSO, the Xilinx XC4000 FPGA which has 276 CLBs in a 16×16 matrix is considered in which the placement of circuit which contains 20 CLBs is taken as our problem.

In PSO, each particle represents a Xilinx XC4000 FPGA with 16×16 CLBs. For the implementation of our problem, those 20 CLBs are randomly placed on the FPGA and are allowed to move within the 16×16 matrix space. The coordinates (row, column) of the 20 CLBs on the FPGA are taken as the position vector of each particle in a swarm. i.e., each particle's position is a matrix of 20×2.

The fitness value or the desired performance function of the particles is evaluated by adding the normalized distances of respected connections between the CLBs where applicable. For example, if there is an interconnection between two CLBs whose locations are (row1, column1) and (row2, column2) respectively, then the fitness function is given by,

$$f = (abs(row1 - row2) + abs(column1 - column2)) \quad (5)$$

The position vector or locations of all the 20 CLBs on the FPGA whose fitness function is the lowest is stored by each particle as pbest and the gbest stores the position vector or the locations of all the 20 CLBs with the lowest fitness function of the particle in the whole swarm. The values of pbest and gbest are updated continuously whenever a position vector with the lowest fitness is found for each particle and the whole swarm respectively. Every pbest is a near optimal position vector and gbest is the global optimal position vector after a number of PSO iterations for the FPGA placement.

For faster convergence, there should be a balance between the both constants .It is also found that the values should not go beyond 2.5, after which PSO becomes unstable due to very high velocity. Simulations have been done varying the values of C1 and C2 in a range of 1 to 2.5 while both increase at the same time and while one increases but the other decreases. And it's found that C1 = C2 = 1.2 gives the best result.

4. RESULT

The FPGA placement for the implementation of our circuit using PSO has a swarm of particles which start with randomly initialized position vectors for the placement of CLBs in the FPGA. Fig.1 shows the position vector of the CLBs corresponding to the initial gbest of the swarm with the fitness of 707 for the values of C1=C2=1.2. A number of trials yielded a fitness value of 158 on average over 10000 explorations. Similarly, the average fitness value over 15000 explorations is also found randomly by changing the values of C1 and C2 and the obtained results are tabulated below.

When PSO is used to choose optimal positions for the CLBs placement, they have been found to be placed almost adjacent to each other. In our experiment, the CLBs positions are restricted from overlapping. If such condition is removed all the CLBs are found to overlap with the pbest and gbest fitness values of the particles and the swarm respectively zero. Therefore, the PSO algorithm finds the global minimum value for all the particles for the FPGA placement. And hence it can be proved that PSO is a population based global optimization technique.

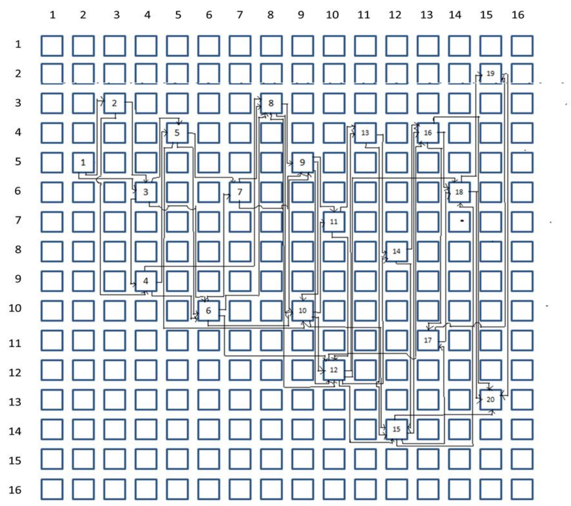

Fig.1. Position vector of the CLBs corresponding to the initial gbest of the swarm

Table1. Values of gbest considering the effects of acceleration constants C1 and C2

Acceleration Constants		No. of iterations = 10000			No. of iterations = 15000		
C1	C2	Initial gbest	Final gbest	Execution time (in sec)	Initial gbest	Final gbest	Execution time (in sec)
2.1	2.4	729	173	13.75	720	207	26.11
1.3	2.2	753	223	20.52	696	191	22.16
1.5	2.0	705	166	10.50	696	219	21.49
1.7	1.8	715	221	16.21	716	211	28.51
1.9	1.6	738	284	13.75	716	209	18.34
2.1	1.4	726	321	20.99	752	299	25.99
2.3	1.2	735	229	13.65	706	232	19.22
2.5	1.0	721	293	17.76	721	263	32.57
1.0	1.0	715	231	13.93	723	177	20.77
1.1	1.1	708	171	9.83	699	191	27.17
1.2	1.2	707	158	11.31	721	180	19.19
1.4	1.4	718	184	18.86	714	198	26.04
1.6	1.6	714	185	12.69	710	263	25.80
1.8	1.8	712	252	17.84	712	228	23.36
2.0	2.0	712	293	19.73	733	282	31.31
2.5	2.5	731	300	20.88	733	301	18.96

Note: We are getting the best result at C1=C2=1.2

5. SIMULATION RESULTS

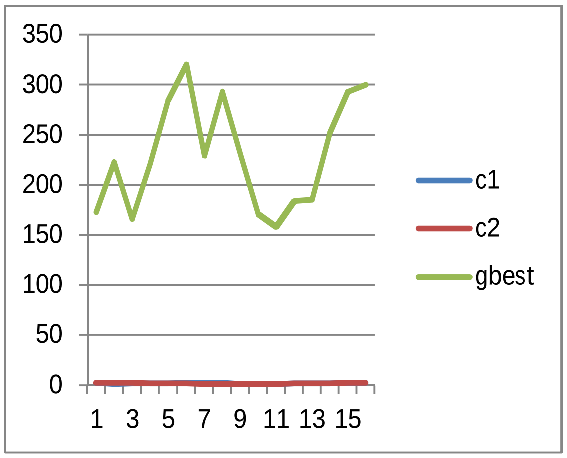

Fig2. gbest w.r.t c1 and c2 for 10000 iterations

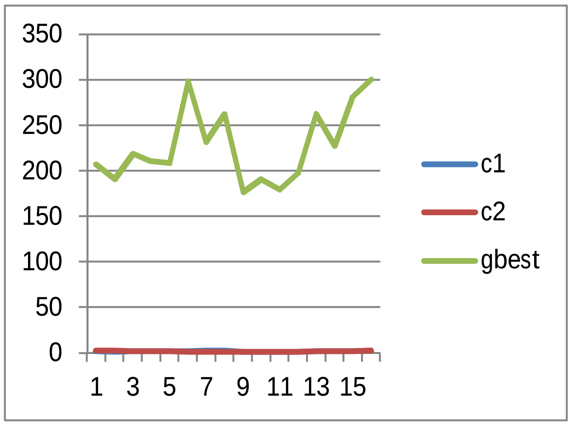

Fig3. gbest w.r.t c1 and c2 for 15000 iterations

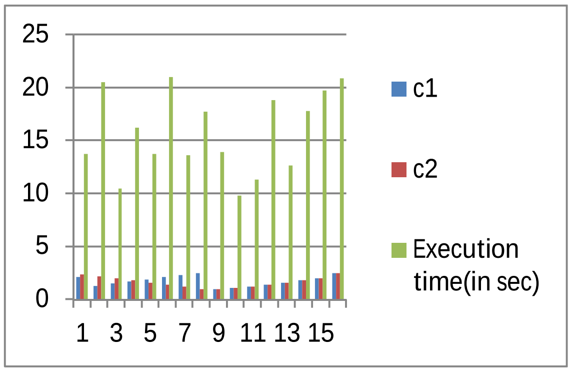

Fig4. Execution time (in sec) w.r.t c1 and c2 for 10000 iterations

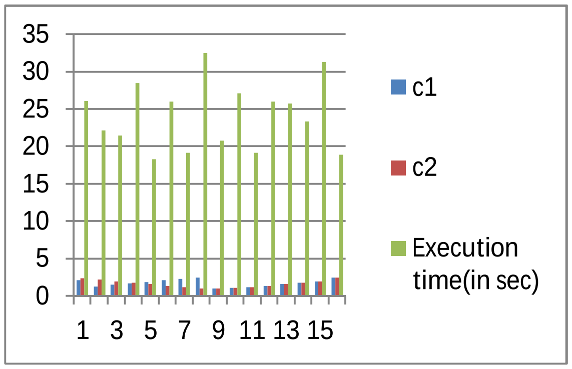

Fig5. Execution time (in sec) w.r.t c1 and c2 for 15000 iterations

6. CONCLUSION AND SCOPE FOR FUTURE WORK

Thus, the interconnection lengths between CLBs in FPGA can be minimized using PSO. FPGA placement using PSO can be optimized with proper choice of acceleration constants. The experiment has been dealt only with the placement problem. After this, routing the interconnection in between the CLBs and I/Os to achieve the desired functionality should be done. Routing is also an NP-Complete class of problem and need to be optimized.

REFERENCES

[1] A.Benuel Sathish Raj , Y.Adline Jancy, K.M.Senthilkumar and M.Sangeetha, “Optimization of Placement and Routing process in XILINX XC4000 FPGA with 14X14 CLBs Using PSO”, International Journal of Emerging Technology and Advanced Engineering Website: www.ijetae.com (ISSN 2250-2459, Volume 2, Issue 7, July 2012)

[2] Bingfeng Mei, Patrick Schaumont Serge VernaldeA “Hardware-Software Partitioning and Scheduling Algorithm for Dynamically Reconfigurable Embedded Systems”, IMEC vzw, Kalpedreef 75, B-3001, Leuven, Belgium Department of Electrical Engineering, K. U. Leuven, B-3001, Leuven, Belgium.

[3] Riccardo Poli, James Kennedy, Tim Blackwell, “Particle Swarm Optimization; An overview” in Springer Science + Business Media, LLC 2007 .10 May 2007

[4] Chen-Chi Chiang, Hardware / Software Real-Time Relocatable Task Scheduling and Placement in Dynamically Partial Reconfigurable Systems, A Thesis Submitted to Institute of Computer Science and Information Engineering College of Engineering National Chung Cheng University for the Degree of Master in Computer Science and Information Engineering, June 2007

[5] Christoph Steiger, Herbert Walder, Marco Platzner, Lothar Thiele, Computer Engineering and Networks Lab, Swiss Federal Institute of Technology (ETH) Zurich, Switzerland, ‘Online Scheduling and Placement of Real-time Tasks to Partially Reconfigurable Devices’.

[6] Venu G. Gudise and Ganesh K. Venayagamoorthy, *Dept. of Electrical and Computer Engineering, University of Missouri – Rolla, MO 65409, USA,* ‘FPGA Placement and Routing Using Particle Swarm Optimization’.