International Journal of Computer Engineering Science (IJCES)
Volume 2 Issue 7 (July 2012) ISSN : 2250:3439
https://sites.google.com/site/ijcesjournal http://www.ijces.com/

Performance Imrovement of a Navigataion System Using Partial Reconfiguration

S.S.Shriramwar[1], Dr. N.K.Choudhari[2]

[1] Priyadarshini College of Engineering, R.T.M. Nagpur Unversity,Nagpur,
sshriramwar@yahoo.com
[2]Smt.B.Chaturvedi College of Engineering, R.T.M. Nagpur Unversity,Nagpur
drnitinchoudhari@gmail.

Abstract. Dynamic Partial Reconfiguration (DPR) of FPGAs presents many opportunities for application design flexibility, enabling tasks to dynamically swap in and out of the FPGA without entire system interruption. In this thesis, we have implemented a line follower robot for the white line as well as for black line, both these modules are programmed in VHDL. The robot are made to run for white line and it will dynamically reconfigure the FPGA in the run-time for the black line or vice-versa. This design includes two modules one is static and the other is partially reconfigurable regions (PRR) which is a dynamic region. The controllers are the static modules used for controlling the flow of data to and from the reconfigurable modules to the external world (host environment) through busmacros. Whereas white line and black line modules are designed as dynamic modules .

Keywords: Reconfiguration, Xilinx, FPGA , Filters

1 Introduction

In the last few decades, Reconfigurable Computing has become popular in the area of computer architectures. Reconfigurable systems arise to compensate the differences of flexible microprocessors and high-speed ASIC circuits. A reconfigurable architecture takes advantages of both systems. It is more flexible than ASIC circuits since it can be reconfigured with changing computing needs. In addition, it has better performance than processors since it implements the desired algorithm on a dedicated hardware.

Reconfigurable architectures take place in between microprocessors and ASICs according to the flexibility and speed. Reconfigurable SoC’s based on Field Programmable Gate Arrays (FPGAs) are, therefore, being designed to meet these requirements. This technology has been popularized in the recent past and there are a number of products based on high density FPGAs. In contrast to standard cells and gate arrays, FPGAs can be easily erased or reprogrammed. The latest version of these FPGAs introduce the concept of ‘Dynamic Run-time Reconfiguration’, where only a small portion of the circuitry is modified at runtime while the system remains functioning [6]. Dynamic partial reconfiguration is going to make hardware more flexible by giving a FPGA the capability to modify its internal structure on the fly deemed ideal for certain applications in real-time manufacturing simulation.

2 Problem Definition.

In the industry carriers are required to carry products from one manufacturing plant to another which are usually in different buildings or separate blocks. Conventionally, carts or trucks were used with human drivers. Unreliability and inefficiency in this part of the assembly line formed the weakest link. The project is to automate this sector, using carts to follow a line instead of laying railway tracks which are both costly and an inconvenience

2.1 Line Follower Robot

Perhaps the simplest navigation system for mobile robots involves following some predefined path that’s marked on the ground. The path can be a black or white line painted on a hard-surfaced floor, a wire buried beneath a carpet, a physical track, or any of several other methods. This type of robot navigation is used in some factories. Marking the path with reflective tape is preferred in factories because the track can easily be changed without ripping up or repainting the floor. You can readily incorporate a tape-track navigation system in your robot. The line tracing feature can be the robot’s only means of semi-intelligent action, or it can be just one part of a more sophisticated machine. You could, for example, use the tape to help guide a robot back to its battery charger nest. With a line-tracing robot, you place a piece of white or reflective tape on the floor. For the best results, the floor should be hard, like wood, concrete, or linoleum, and not carpeted. One or more optical sensors are placed on the robot. These sensors incorporate an infrared LED and an infrared phototransistor

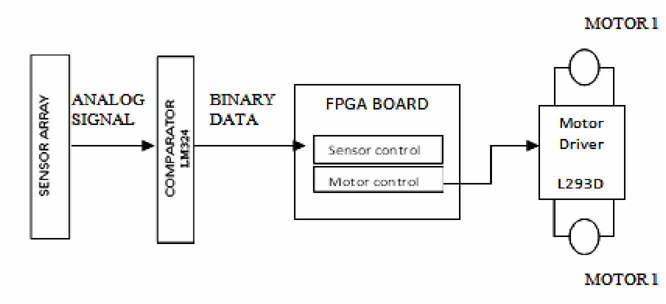

Fig1 - Block diagram of line follower robot

3. Module-Based Partial Reconfiguration

The modular design flow allows the designer to split the whole system into modules. For each module, the designer generates a configuration bit stream. starting from an HDL description and going through the synthesis, mapping, placement, and routing procedures, independently of other modules. The modular design flow consists of „Modular Design Entry/Synthesis and Modular Design Implementation., Modular Design Entry and Synthesis step must be done for top-level design and the modules. Top-level design is designed by team leader and consists of black box for each sub-modules and wiring for interconnection of each sub-modules.

Column Based Reconfiguration: As mentioned before, Xilinx FPGAs give an opportunity that a column of Configurable Logic Blocks (CLBs) can be reconfigured by writing its belonging frames to a configuration port of the FPGA. This structure and additional features enable creating Runtime Reconfigurable (RTR) architecture on Xilinx FPGAs. FPGA can be divided into multiple columns to make a RTR system. By using Xilinx map, place and route tools it is possible to generate bitstream for only one reconfigurable column. Then columns may be reconfigured by this bitstream while the other columns are still working. This reconfiguration operation is called active partial reconfiguration of FPGA..Here reconfigurable modules communicate with other modules through bus macros. An example partial reconfigurable architecture with two reconfigurable modules is shown in Figure.

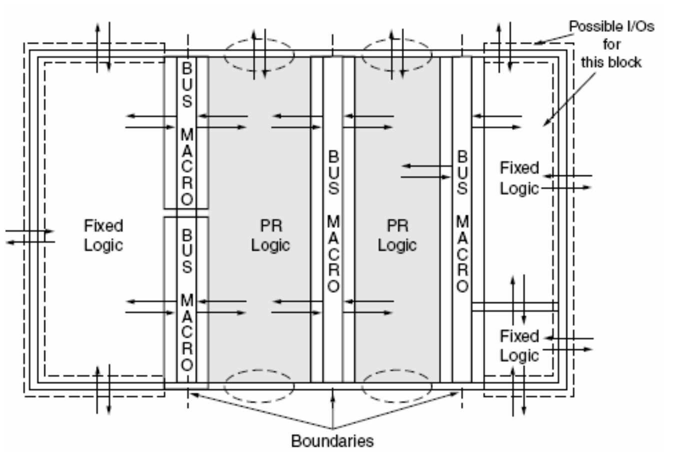

Figure.2. Design Layout with Two Reconfigurable Modules

The tri-states bus macros for Virtex, Virtex-II, Virtex-E, Virtex-II pro, Spartan 2, Spartan 2E can be downloaded from the zip file attached to Xilinx application note XAPP290 . Sometimes tri-states buffers bus macros cannot be used, as in the Spartan FPGAs which lack tri-states buffers so a slice bus macro can be implemented instead:

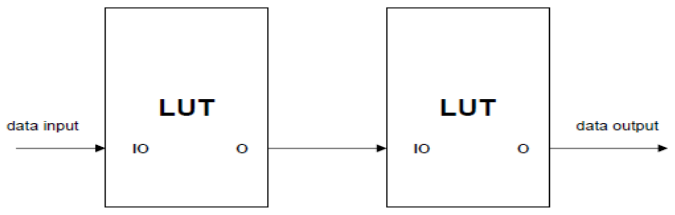

Fig.3 Slice bus macro

Initial budget phase: In this phase, the team leader assigns top-level constraints to the top-level design. Top-level constraint needs to area constraint and bus macro assignment.

Active module implementation:

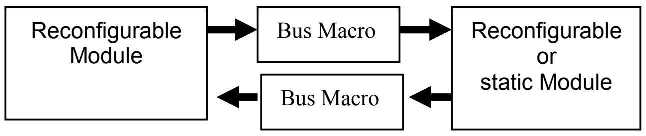

Figure.4. Communication with Reconfigurable Modules

4 Final Assembly

The final assembly phase is the process of combining each of the individual modules back into a complete FPGA design. The placement and routing achieved during the active implementation phase for each module will be preserved, thereby, maintaining the performance of each module.

So we can perform a unique assembled design and then changing the partially reconfigurable modules with their bitstream. We implement the assembled design using the same flow as in the active module phase.

5 Pulse width modulation

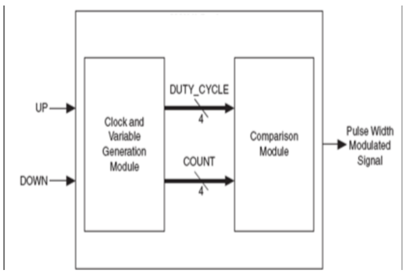
UP
DOWN
Clock and Variable Generation Module
DUTY_CYCLE
4
COUNT
4
Comparison Module
Pulse Width Modulated Signal

Figure 5 Implementation of PWM

Pulse-Width Modulation (PWM) is used in many diverse areas such as controlling the speed of DC motors, and the analog translation of digital audio using the 1-bit DAC of a CD player. The design presented in this brief uses a register to store the desired ‘mark’ value, which is automatically loaded into a down counter upon reaching its terminal count. The PWM Frame Period is the product of the counter’s clock period and the terminal count value, being the sum of the ‘mark’ and ‘space’ periods. The design is readily scaled up or down simply by changing the width of the register and counter.

6 Module based partial reconfiguration

Module-based partial reconfiguration method is a special case of modular design. And this method can reconfigure only a given subset of internal components during device is activating. A complete initial bit stream must be generated, and then, partial bit steams are generated for each reconfigurable module. Fig. 6 shows the design flow of module-based partial reconfiguration. Hardwired Bus Macros must be included in design as shown in Figure 6.

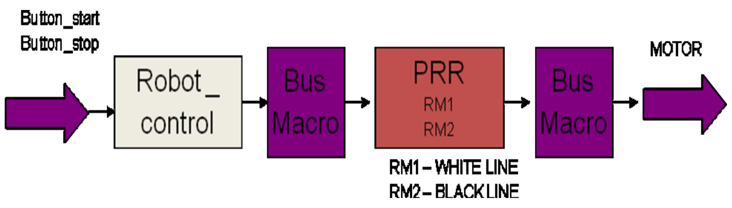

Figure.6. Design to be implemented on plan ahead

PR flow using Plan Ahead

step 1 – Create Plan Ahead Project Step 2 – Create Static and RM Pblocks
Step 3 – Place Components Step 4 –Set and Add RMs
Step 5 –DRC Check and View Information
Step 6 –Run PR Implementation Flow Step 7 –Testing

7 Results

Figure 7 shows the simulation results of control module which was implemented in modelsim. Table 1,2 and 3 shows the device utilization , memory utilization and power consumption with and without partial reconfiguration.

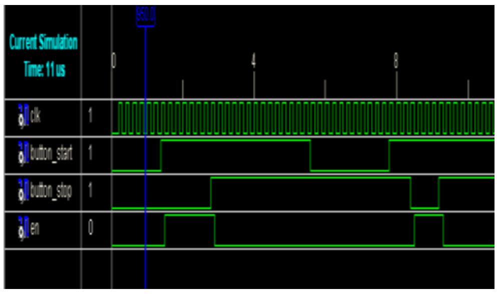

Figure.7 Simulation of Control Module

Table 1: Device Utilization with and without PR

Logic Utilisation	Used With PR	Used without PR	% Saving
No. of slice flip flops	3120	6086	48.73
No of 4 input LUTs	4166	9091	58.17

Table 2: Memory Utilization with and without PR

Design	file size (in kb)	Reconf. time(in sec)	frames
RM 1	130	0.68	1339
RM 2	143	0.78	1352
Static	780	2	3095

Table 3: Power consumption with and without PR

Power	Used With PR	Used without PR	% Saving
Power dissipated (W)	3.45	2.85	17.39

8 Conclusion

From the results of table 1, 2 and 3 , it is clear that using the partial reconfiguration device area ,memory and the power consumption can be reduced to improve the performance of a navigation system.

References

1. Emi Eto, "Difference-Based Partial Reconfiguration", XAPP290 (v2.0) December 3, 2007.
2. Virtex Series Configuration Architecture User Guide, Xilinx Application Note XAPP151, version 1.1, Xilinx, Inc. ,1999.
3. Virtex –4 platform FPGA User Guide, version 2.6, Xilinx,Inc., 2008.
4. Two Flows for Partial Reconfiguration: Module Based or Difference Based, Xilinx Application Note XAPP 290, version 1.2, Xilinx Inc.2004.
5. S.Commuri, V.Tadigotla, L.Sliger " Efficient Controller implementations . for Robot Control " Circuits, Systems, Electronics, Control & Signal . Processing, Dallas, USA, November 1-3, 2006
6. P. Sedcole, B. Blodget, T. Becker, J. Anderson and P. Lysaght "Modular dynamic reconfiguration in Virtex FPGAs " IEE Proc.-Comput. Digit. Tech., Vol. 153, No. 3, May 2006
7. Early Access Partial reconfiguration User Guide For ISE 9.2.04i UG208 (v1.2) September 9, 2008
8. P.K.Jawahar , V.Vaidehi "Analysis of reconfigurable Architecture for Delay Sensitive Voice Streams Over IP Networks", Asian Journal of Information Technology 5(12) : 1458-1463, 2006
9. Khaled Benkrid," High Performance Reconfigurable Computing:From Applications to Hardware" IAENG International Journal of Computer Science, 35:1, IJCS_35_1_04
10. Yan Meng , Kerry Johnson, Brian Simms, and Matthew Conforth" A Modular-based Miniature Mobile Robot for Pervasive Computing" International Journal of Hybrid Information Technology Vol. 1, No. 1, January, 2008.
11. Xilinx, Inc., "Using a Microprocessor to Configure Xilinx FPGAs via Slave Serial or SelectMAP Mode", November 2003.
12. Virtex –5 FPGA User Guide UG190 (v5.3) Xilinx,Inc., May 17, 2010