A New Technique of Embedding Multigrain Parallel HPRC in OR1200 a Soft-Core Processor

MAHESWARI.R[1] & PATTABIRAMAN.V[2]
School of Computer Science & Engineering Department
VIT University
Kelambakkam- Vandalur Road , Chennai 600048
INDIA
1.maheswari.r@vit.ac.in 2. pattabiraman.v@vit.ac.in

Abstract: - In the embedded era, reconfigurable components comes in three forms of IP Intellectual Property cores i)Soft core ii) Firm core and iii) Hard Core. This paper presents a new technique of embedding multigrain parallel processing HPRC using FPGA in the CPU/DSP unit of OR1200 a soft-core RISC processor. The core performance is increased by placing a multigrain parallel processing HPRC internally in the Integer Execution Pipeline unit of the CPU/DSP core. Depending on the complexity/depth of the code, the dependency level of vertices DL were created and numbers of threads N were created to run the code parallel in HPRC. Multigrain parallel processing HPRC is achieved by two function i) HPRC_Parallel_Start to trigger the parallel thread ii) HPRC_Parallel_End to stop the thread. In the first phase of this paper a Verilog HDL functional code is developed and synthesised using XIINX ISE and in the second phase a CoreMark processor core benchmark is used to test the performance of the reconfigured IP soft core.

Key-Words: - IP Core**,** Soft-core processor, OR1200, FPGA, ASIC, HPRC, Benchmark

1 Introduction

Embedded system is any dedicated or special purpose electronic system that contains microcontroller or microprocessor to carry out specific tasks. An Embedded system integrates application specific hardware, embedded software and real time operating system into functional unit. Based on the component used in the embedded system the embedded processors are classified into two major division .i) Micro-Controller (μC) based Embedded Processor ii)Micro-Processor (μP) based Embedded Processor. Reconfiguring these components will increase the overall processor performance.

1.1 Intellectual Property Core

An Intellectual Property (IP) cores are any logic block which is predefined and reusable array of structure. This IP cores are reconfigurable component in any synthesizable programmable hardware unit like FPGA (Field Programmable Gate Array) and ASIC (Application Specific Integration Circuit). These reconfigurable component comes in three forms of IP cores such as i)Soft core (microprocessor implemented using programmable logic) ii) Firm core (microprocessor implemented using gate-level net-list) and iii) Hard Core (microprocessor implemented using component layout).

1.2 Soft-Core processor

A soft core processor is a reconfigurable/reusable embedded computing in HDL (Hardware Description Language) which can be synthesized in any programmable hardware unit like FPGA (Field Programmable Gate Array) or ASIC (Application Specific Integration Circuit).The soft-core CPU is used to create an FPGA-based system-on-chip (SoC). The merits of using softcore processor are 1. higher level of abstraction (easier to understand) 2. more flexible (core can be changed by editing the source code or selecting parameter) 3. platform independent (synthesizable in FPGA, ASIC etc).

RISC based embedded soft-core CPU processor comes in open source cores and commercial cores. The various open source soft core processor available under LGPL (Lesser GNU Public License) and GPL (GNU Public License) are OpenRisc 1200 (LGPL), S1 core (GPL), Leon 2 (LGPL),Leon 3 (GPL), OpenFire (MIT), AeMB, PacoBlaze (BSD), DSPuva16 ,LatticeMico32, LatticeMico8 etc. Nios II (f/s/e), PicoBlaze Cortex-M1, MicroBlaze, Xtensa are the commercial FPGA based soft-core processor recently offered by Altera , Xilinx ,Tensilica etc .

FPGA is an IC(Integrated Circuit) which consists of array of configurable logic block. These logic blocks can be programmed after manufacturing. The main advantages of FPGA's are 1. Flexible development of functional blocks 2.Reduced time and risk factor in development of model 3. Market in time 4. Long time availability of code.

1.2.1 Soft-Core in FPGA

FPGAs with soft core processor provide configurable design with increased flexibility[1]. The various configurable parameters are 1. Instantiating cache 2. Instantiating data path unit 3. Instantiating instruction set [2].

An FPGA is used for HPRC (High Performance Reconfigurable Computing) due to its flexible array programmable configuration in "logic cells". The intrinsic parallelism of FPGA allows the high performance reconfigurable computation even at very low clock rates. In HRPC, FPGA serves as coprocessor to the microprocessor in which the system invokes the appropriate FPGA architecture to execute the target operation [3][4][5][6]. These FPGA devices are particularly suitable for parallel processing. HPRC has three major challenges to overcome. They are 1. Programmability 2. Memory bandwidth performance 3. Price/ performance [4].

2 Existing OR1200 architecture

OpenRISC 1200(OR1200) is a 5 stage integer pipeline(Fetch, Decode, Execute, Memory and Writeback) IP core CPU processor available in open source under Lesser GNU Public License (LGPL) .OR1200 RISC based soft core is used in embedded, automobile, internet, networking, telecommunication, portable and wireless applications. The OR1200 model implements ORBIS32 Instruction set with a 32-bit processor conforming to Harvard micro-architecture (separate Instruction and Data cache). The OR1200 core's architecture [7] Fig.1. consists of CPU/DSP, Memory management unit -separate Data MMU & Instruction MMU, Power Management Unit, Advance Debug unit (Debug I/F), Integrated Ticker Timer, Programmable Interrupt Controller (Interrupts), Instruction Cache, Data Cache, Wishbone Interface (a SoC interconnection architecture for portable IP core .It is a logic bus available as open source hardware bus).

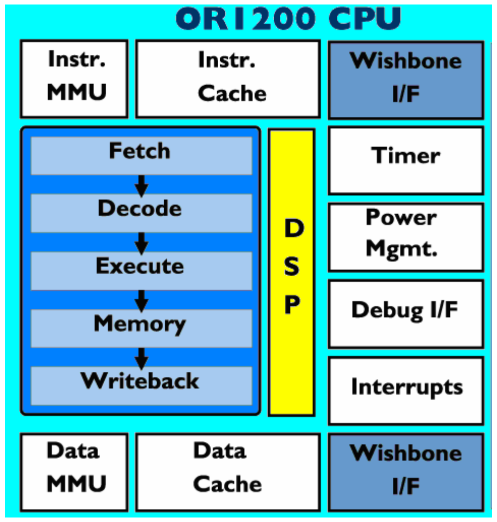

Fig.1. OR1200 Core's Architecture [7]

3 Proposed Technique

In this paper, as a first phase a functional code for a reconfigurable multigrain parallel HPRC is implemented inside CPU/DSP (Integer Execution Pipeline unit) of OR1200 a soft core RISC processor.

3.1 Integer Execution Pipeline unit with HPRC

The OR1200 core implements the following types of 32-bit integer instructions:

1. Arithmetic instructions
2. Compare instructions
3. Logical instructions
4. Rotate and shift instructions

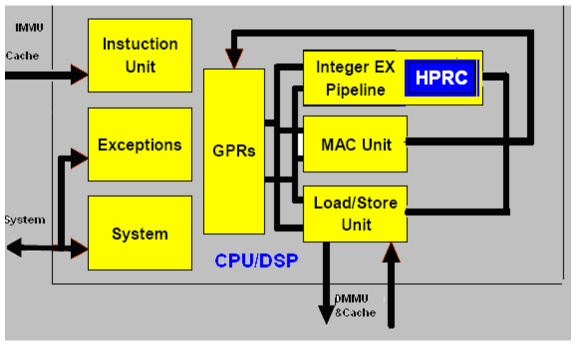

Fig.2. CPU/DSP Block diagram with embedded HPRC in Execution unit

An embedded parallel processing HPRC in OR1200 Integer Execution pipeline unit is shown in Fig.3. The performance of HRPC unit inside the OR1200 soft core is achieved by dynamic hardware multitasking [1].Through ICAP (Internal Configuration Access port) the inter-module communication is ensured which is a reasonable medium for cross chip communication [5] & [6].

To encompass parallelism in HPRC the DOP (Degree of Parallelism) has to be identified in the execution unit. In [8] iteration level parallelism, depending upon the DOP, the execution of the loop can be categorized into three ways 1.serial loop 2. parallel loop (DOALL) 3. partially parallel loop (DOACROSS).

3.2 Multigrain parallelism

Using 2D IDCT and lifting DWT a trade off between the DOP (Degree of Parallelism) and NoP(No of Operation) for a reconfigurable architecture(both fine and coarse grained) is exploited in an intrinsic and platform independent parallelism algorithm at various granularity namely multigrain parallelism .A robust frame work was proposed which capable of quantifying parallelism in both the algorithms and architectures using linear algebra[9][10].In this paper IOS(Independent Operation Set) is identified from the given instruction set and depending on the complexity/depth of the code, the dependency level of vertices DL were created and numbers of threads N were created to run the code parallel in HPRC for SIMD (Single Instruction Multiple Data)[9].

The multigrain parallelism in HPRC is accomplished by two functions i)HPRC_Parallel_Start - to trigger the parallelism ii)HPRC_Parallel_End - to stop the parallelism. In this phase, a reconfigurable [11][12][13][14] Verilog HDL functional code for multigrain parallelism [15][16] i.e internal parallelism[17] inside integer execution unit of the IP core is implemented, synthesized and simulated in XILINX ISE with following configuration (Table 1).

TOOL	**XILINX ISE**
FAMILY	**SPARTAN3**
DEVICE	**XC3S200**
PACKAGE	**FT256**
SPEED	**-4**
SIMULATOR	**ISE Simulator (ISim)**

Table 1 .Configuration Details

The pseudo code for the multigrain parallel HPRC is given in the Fig.3.

```
BEGIN
Identify the IOS
WHILE SIMD
  HPRC_Parallel_Start
  Set the dependency level for leaf node $DL_1$
            $DL_1 = 1$
  Set the max depth M= max(DL +1)
DO
            Initialize the granularity size k=1
            Set the dependency level for leaf node
            $DL_1$, lower bound (DLL) and upper bound
            (DLU)for all vertices
            DLL =1
            DLU = DLL + K – 1
            Set the max depth M= max(DL +1)
WHILE DLU reaches the leaf node
 increment
            k=k+1
            DLL= $DL_1$ +1
            DLU= $DL_{M+1}$
HPRC_Parallel_End
END WHILE
END
```

Fig.3 Pseudo code- multigrain parallel HPRC

The Verilog HDL code for reconfiguring OR1200 core for HPRC is synthesized for generating the gate level netlist .The Fig.4 below shows the RTL (Register-Transfer Level) schematic for multigrain parallel HPRC generated in XILINX ISE.

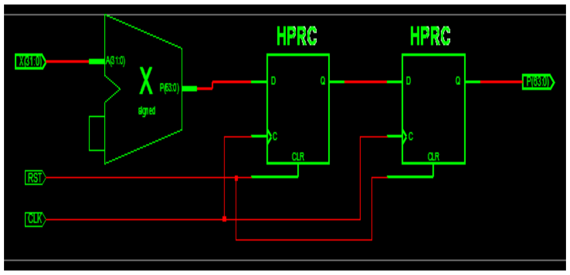

Fig.4 View RTL Schematic for embedded HPRC in Integer Execution unit

3.3 Parallel Computation

Parallel computation for any processor is calculated using Amdahl's law, which states that the speed of the processor is inversely proportional to the non-parallelized part (sequential part) α of the program. The equation is given by

$$\text{speedup} = t_1/t_p \qquad (1)$$

$$s= (m/r_1*10^6)/((m*q/P1*p)+(m*(1-q)/P1)) \quad (2)$$

In equation (1) t_1 is the serial execution time of processor & t_p parallel execution time of processor .Using equation (1) in a system if the sequential part of the program is 15% of the entire code then the overall speed up of the system is increased to 15 times. In equation (2) s is the system speedup, $P1= r_1*10^6$,m is the number of operations in the program , p is the processing element executed in parallel , q is the fraction of the parallel operations can be executed in parallel, and 1-q operate in serial mode. In equation (2) when q=1(i.e fully parallelized code) the speed will be very high 100% which is highly difficult and even q= 0.9 to 0.95 is very difficult to obtain. This technique ensures the value nearing q=0.5 and p=1024 attains the maximum performance has 1.998049

In [11] the application program is translated into parallel hardware threads then those threads are executed by reconfigurable computing. Fig.5 ensures optimizing the state machine of FSM (Finite State Machine) with Johnson encoding on signal states from 0:3 generated while synthesizing the functional code.

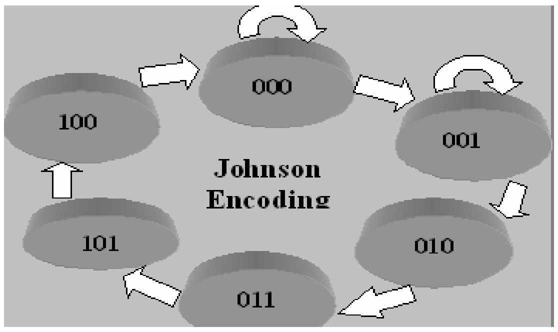

Fig.5 OR1200- Optimizing-FSM with Johnson encoding on signal <state[1:3]>

4 Results and Discussion

Benchmark provides a method of comparing the performance of chip/system architecture. Dhrystone is most commonly used synthetic benchmark which is open source, small, easily portable and is much used as compiler benchmark rather than hardware benchmark i.e more compatibility towards compiler. In this paper CoreMark bench mark is used to test the performance of the multigrain parallel HPRC embedded in OR1200 processor core rather than classical Dhrystone benchmark.

4.1 CoreMark BenchMark

CoreMark is a embedded generic benchmark developed by EEMBC (Embedded Microprocessor Benchmark Consortium) specially for processor's core features [18][19]. The graph CoreMark Vs Dhrystone Benchmark in Fig.6 shows the efficiency and compatibility of CoreMark towards processor core.

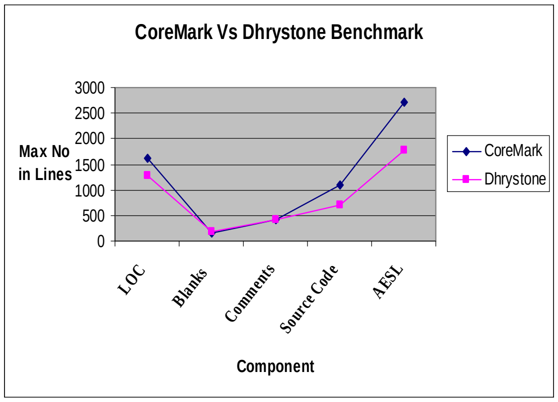

Fig.6 CoreMark Vs Dhrystone Benchmark

To test the reconfigured multigrain parallel HPRC the matrix manipulation test bench in CoreMark is used. Since Matrix manipulation is very simple algorithm forms the basis of many more complex algorithms. The tight inner loop is the focus of many optimizations (compiler as well as hardware based) and is thus relevant for embedded processing. The Table 2 shown below signifies the 2K performance run parameters for CoreMark benchmark compiled in Compiler version-GCC3.4.4 and Compiler flags -O2

Run type	**Memory Location**
seedcrc (identifier for the input seeds)	0xe9f5
crclist (validation for matrix part)	0x1fd7
crcstate (validation for state part)	0x8e3a
crcfinal (iteration dependent output)	0x33ff

Table 2 .Performance run parameter

4.2 Output Simulation

In the second phase of this paper the developed functional code for multigrain parallel HPRC is simulated to analysis its core performance. Fig.7 shows the behavioural model simulation of HPRC with multigrain parallelism tested for Matrix manipulation test bench [14] using CoreMark benchmark in XILINX ISE Simulator (ISim).

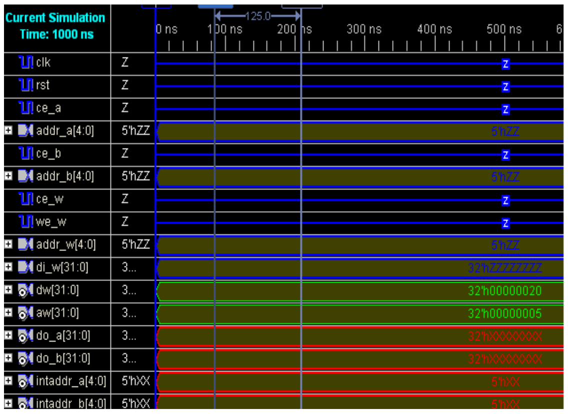
Current Simulation Time: 1000 ns
125.0
0 ns
100 ns
200 ns
300 ns
400 ns
500 ns
6
clk Z
rst Z
ce_a Z
addr_a[4:0] 5'hZZ
ce_b Z
addr_b[4:0] 5'hZZ
ce_w Z
we_w Z
addr_w[4:0] 5'hZZ
di_w[31:0] 3... 32'hZZZZZZZZ
dw[31:0] 3... 32'h00000020
aw[31:0] 3... 32'h00000005
do_a[31:0] 3... 32'hXXXXXXXX
do_b[31:0] 3... 32'hXXXXXXXX
intaddr_a[4:0] 5'hXX
intaddr_b[4:0] 5'hXX

Fig.7 Behavioral Model Simulation in ISim

4.2 Performance Analysis

The simulated model for the test bench program is analyzed for the performance of the core OR1200.

OR1200	Existing	Proposed
Total ticks	25875	25875
Total time (secs)	25.875	12.875
Iterations/Sec	3864.7343	1468.6003
Iterations	100000	70990

Table 3 .Configuration Details

Table 3 (Configuration Details) signifies the performance comparison of the existing core (OR1200 without HPRC) with the proposed one (OR1200 with multigrain parallel HPRC) considering the Matrix manipulation program in CoreMark benchmark. In this analysis the number of Iterations is reduced to approximately 30% and there by the iteration per sec is reduced to approximately 38%. Fig.8 shows the performance analysis graph drawn for the above configuration table.

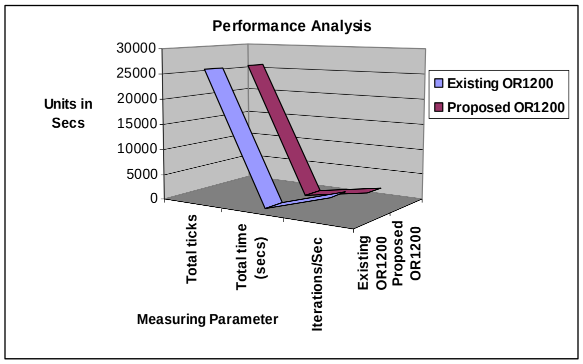

Fig.8 Performance Analysis of OR1200 using CoreMark Benchmark

5 Conclusions

In this paper, a new technique of reconfigurable multigrain parallel HPRC is proposed to increase the performance of OR1200 a soft core processor in the embedded system. In the first phase a Verilog HDL code is developed and implemented using XILINX ISE to synthesize HPRC in Integer execution unit of OR1200. In the second phase, a behavioral model simulation is generated for CoreMark benchmark matrix manipulation and the performance of the reconfigured architecture is identified using ISim.

Thus the proposed technique of reconfigurable multigrain parallel HPRC can be synthesized into the FPGA gate logic. This technique can be incorporated in the various open source soft core IP cores to increase the overall core performance with very less overhead in hardware change.

References:

[1] Peter Borisov Minev ,Valentina Stoianova Kukenska “Implementation of soft-core processor in FPGAs”,*UNITECH’07 International Sceintific Conference*,2007

[2] Chuan Hong, Khaled Benkrid, Xabier Iturbe, Ali Ebrahim, and Tughrul Arslan “Efficient On-Chip Task Scheduler and Allocator for Reconfigurable Operating Systems”*IEEE Transaction Embedded Systems Letters*, Vol. 3, No. 3, pp:85-88, 2011.

[3] Tarek El-Ghazawi, Esam El-Araby, and Miaoqing Huang,Kris Gaj, Volodymyr Kindratenko,Duncan Buell ” The Promise of High-Performance Reconfigurable Computing

" *IEEE Computer Society February*, 2008 pp:9-76

[4] Mark Howison, E. Wes Bethel and Hank Childs "Hybrid Parallelism for Volume Rendering on Large-, Multi-, and Many-Core Systems "*IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS*, VOL. 18, NO. 1, JANUARY 2012. This article has been accepted for inclusion in a future issue of this journal.

[5] G. Tan, N. Sun, and G. R. Gao, "Improving Performance of Dynamic Programming via Parallelism and Locality on Multicore Architectures," *IEEE Trans. on Parallel and Distributed Systems*, pp. 261-274, vol. 20 No.2, Feb., 2009.

[6] Rob Baxter1, Stephen Booth, Mark Bull, Geoff Cawood, Kenton D'Mellow, Xu Guo, Mark Parsons, James Perry, Alan Simpson, Arthur Trew " High-Performance Reconfigurable Computing – the View from Edinburgh**"** *IEEE publication* in press

[7] OR1200 http://www.opencores.org

[8] Duo Liu, Yi Wang, Zili Shao, Minyi Guo, and Jingling Xue," Optimally Maximizing Iteration-Level Loop Parallelism" *IEEE Transaction on Parallel and distributed system*,2011.

[9] Gwo Giun Lee, He-Yuan Lin, Chun-Fu Chen, and Tsung-Yuan Huang "Quantifying Intrinsic Parallelism Using Linear Algebra for Algorithm/Architecture Co-Exploration "*IEEE Transaction on Parallel and distributed system*,2011

[10] George Afonso, Rabie Ben Atitallah Alexandre Loyer, Jean-Luc Dekeyser,Nicolas Belanger,Martial Rubio"A prototyping environment for high performance reconfigurable computing"*IEEE Diigtal library*

[11] X. Iturbe, K. Benkrid, T. Arslan, R. Torrego, and I. Martinez, "Methodsand Mechanisms for Hardware Multitasking: Executing and Synchronizing Fully Relocatable Hardware Tasks in Xilinx FPGAs," in *Proc.Int. Conf. Field-Program. Logic Appl.*, Crete, Greece, 2011.

[12] Damjan Lampret "OpenRISC 1200 IP Core Specification" in press

[13] Kosei Shimoo*, Akin Yamawaki' and Masahiko Iwane " A New Parallel Processing Paradigm Using A Reconfigurable Computing System" *lnternational Symposium on Communication and Information Technologie*s 2004 (**ISClT** 2004)

[14] L. Zhuo, and V. K. Prasanna, "Scalable and Modular Algorithms for Floating-Point Matrix Multiplication of Reconfigurable computing Systems," *IEEE Trans. on Parallel and Distributed Systems*, pp. 666-681, Vol. 19, No.5, May, 2008.

[15] A. Brandon, I. Sourdis, and G. N. Gaydadjiev, "General purpose computing with reconfigurable acceleration," *in International Conference on Field Programmable Logic and Applications, FPL*, Los Alamitos, CA,USA, 2010.

[16] Hans-Peter Rosinger, "Connecting Customized IP to the MicroBlaze Soft Processor Using the Fast Simplex Link (FSL) Channel."*in press*.

[17] Seon-yeong Park, Euiseong Seo, Ji-Yong Shin, Seungryoul Maeng and Joonwon Lee "Exploiting Internal Parallelism of Flash-based SSDs", *IEEE COMPUTER ARCHITECTURE LETTERS*, VOL. 9, NO. 1, JANUARY-JUNE 2010

[18] Target http://www.retarget.com/

[19] CoreMark http://www.coremark.org