Journal of Theoretical and Applied Information Technology
30th November 2017. Vol.95. No 22

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

RESOURCES OPTIMIZATION METHODOLOGY FOR HETEROGENEOUS COMPUTING SYSTEM

[1]MAHENDRA VUCHA, [2]DVS CHANDRA BABU, [3]ARVIND RAJAWAT, [4]KARTHIK R

[1,2,4] Department of Electronics and Communication Engineering, MLR Institute of Technology, Dundigal, Hyderabad, India.

[3]Department of Electronics and Communication Engineering, Maulana Azad National Institute of Technology, Bhopal, India.

E.Mail: [1]mahendra.1548@gmail.com

ABSTRACT

Nowadays embedded systems are being equipped with one or more processing cores to support parallel processing of applications and meets design goals at optimized resource utilization. Optimization of resources utilization can be achieved by estimating application requirements in terms of many aspects like performance, resource usage, memory usage, energy consumption, cache performance etc. Among these aspects, estimation of resources is an important to boost execution speed of an application with optimal resources utilization. So, ever increasing system and application complexities made the resources estimation is quite necessary to optimize resources utilization for an application. This paper addresses an optimized heterogeneous computing platform called Heterogeneous Reconfigurable Computing System (HRCS) and also a resources profiling methodology to estimate HRCS resource required for multifarious real life application. The HRCS has been modeled on a single chip Virtex-5 FPGA device and it has been equipped with multiple Reconfigurable Logic Units (RLUs) in combination with a softcore processor as Processing Elements (PEs).

Keywords: *Reconfigurable Logic Unit, Heterogeneous Reconfigurable Computing System, System on Chip, Design Space Exploration.*

1. INTRODUCTION

As the Moore's law demonstrated, the chip manufacturing technology enables the methods for integration of more and more complex functionalities on a single chip called Systems on Chip (SoC). Nowadays the SoCs are utilized to develop advanced embedded systems for the applications like set-top boxes, digital LED televisions, 3G/4G mobile phones etc. The growing technology enabled the customer requirements and created demand to the techniques that satisfy customer needs in less span of time. A digital wave [1] [2] showed in figure 1 by the digital consumer products demonstrates three big waves which could drive electronics industry.

Since the digital consumer products demand high degree of performance, flexibility and personalization, the requirement for high speed computing architectures and technologies has been increased in order to contribute for rising of second digital wave.In generally, the computing systems have many hardware resources like registers, memory, arithmetic and logic units, input/output devices, etc., which are required to process complex problems. The widely used computing system architectures in data processing are classified into four categories:

1. General Purpose Processor (GPP)
2. Application Specific Integrated Circuit (ASIC)
3. Application Specific Instruction Processor (ASIP) such as DSP, Math Co-processor
4. Reconfigurable Architecture (RA)

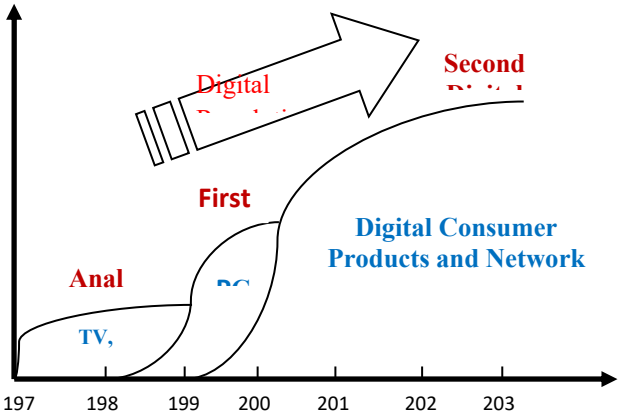

Figure 1: Second Digital Wave [14]

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The software processor system i.e. General Purpose Processor (GPP) brings flexibility while hardware computing system i.e. Reconfigurable Architecture (RA) enhances the performance in addition to flexibility. The emerging data processing technology and multi-core architectural enhancements would play critical role in optimizing the performance of real time applications. The homogeneous multi-core computing systems having similar PEs [11] [12] [15] brings parallel processing to real life applications at the expense of similar resources whereas heterogeneous multi-core computing systems have dissimilar PEs [5] [9] [10] to support applications with dissimilar communication protocols between heterogeneous resources. The reconfigurable system having FPGA as PE [2] [3] [6] [7] bring phenomenon of dynamic reconfiguration to the applications at the overhead of off-chip soft core processor interface. But, the high speed computing systems for multifarious real time applications demand mixed software-hardware systems and it has received significant attention from both academia and industry in recent years.

1.1 Heterogeneous Reconfigurable Computing System

Today most of the System on Chips (SoCs) being realized by using higher end FPGAs as reconfigurable computing area and soft-core processors as software computing unit. The SoC architectures can be equipped with hardware PE, software PE and communication protocols which are very common and necessary for design of real time computing system. In this paper, a hybrid SoC havebeen developed as computing system for minimizing off-chip communication overheads and also to supportexecution of hardware tasks and software tasks as well. The developed hybrid computing system called Heterogeneous Reconfigurable Computing System (HRCS) [8] [13], having integrated softcore processor and multiple Reconfigurable Logic Units (RLU) acts as hardcore processor configured on a single chip FPGA, minimize inter process communication overheads and also have flexibility to support execution of both software tasks as well as hardware tasks of an application. The HRCS provides optimal computing trade-off in execution of software tasks and hardware tasks in terms of performance and area.

1.2 Hardware Software Co Design

Computing systems need to be modeled with dissimilar multiple components to execute significant multifarious function. Nowadays most of the systems are either electronic in nature or embedded with an electronic subsystem for its functionality, control and monitoring. However, the electronic systems and subsystems can be modeled using predominant digital components. Since the majority systems are programmable in nature and also equipped with hardware and software PE, the system performance can be estimated only when they are related to a specific application. The performance parameters of a computing system would be speed, size of the design, manufacturing cost, and flexibility of programming. So, the effectiveness and performance of a system depends on architecture of hardware and software PE of the system. The Hardware Software Co-design methodologies demonstrates the ways of meeting system requirements by exploiting hardware PE and software PE resources effectively for a given set of applications [16]. The HW/SW co-design methodology promises an optimal architecture for a given application and it can be achieved by exploring the design requirements of the applications before building them on a real hardware prototype. The HW/SW co-design is a complex discipline that deals with advances in the areas like computer architecture, VLSI design and software design compilation. Thus co-design methodology is completely depends on the system architecture, application domain, design methodology and implementation technology.

1.3 Hardware Software Partition

Partitioning of design specifications into hardware part and software part is a method that shows major impact on hardware cost and also on performance of the final product. Thus the decisions on hardware and software specifications can be performed by a designer or a CAD tool while considering the architecture and functionality of overall system [16]. The hardware software partitioning decisions always depends on the design specifications. The partitioning problem can be achieved based on the partitioning goals, design specifications, architectural assumptions, and solution strategy.

1.4 General Co-design Problems

The major goal of hardware software co-design is to identify the critical sections of software programs and mapping them effectively to run on the hardware [16]. In order to design cost optimized systems, it is necessary to estimate the design attributes like performance of the software code and area of hardware component. So, the profiling tools are designed to explore design specifications and that helps to estimate the performance of a given

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

application. Profiling and Simulation are to predominant techniques to acquire design and performance parameters of given applications. The simulation offers a high level of flexibility to verify the functionality, where as the profiling supports fast estimation of resources to achieve required accuracy. There are many methods and approaches developed in recent days to estimate design specifications of embedded systems. However the paper has been intended to demonstrate an efficient Design Space Estimation (DSE) methodology to estimate design specifications for real time applications on HRCS. The DSE supports design of optimal computing system for an application and also it helps the designer in rapid prototyping, system integration and optimization of system resources as stated below.

Rapid prototyping: DSE helps the designer in generation of various possible set of prototypes for embedded systems before their actual implementation. Simulation and profiling of prototypes can increase understanding level of the design functions and decisions while considering complex system specifications into account.

Optimization: DSE provides design metrics of multiple implementations of a system design and these metrics can be used to compare one design with another and select optimized design while eliminating inferior designs.

System integration: System integration is the process of assembly and configuration of multiple components into a working model. DSE helps in finding the efficient assemblies and configurations to achieve a required set of design constraints.

2. LITERATURE REVIEW

This section presents overview of the design space exploration methodologies adopted by researchers for hardware software co-design and hardware design space exploration.

2.1 Design Space Exploration

Design Space Exploration (DSE) is a method adopted to implement, analyze and estimate various implementation alternatives in order to find optimal solution. Traditionally, the system designers start with extraction of the required specification of the given design and describe the specifications in computer programming languages such as Verilog HDL, VHDL, and C and then produce a reference executable for implementation of the design. The developed executable is further verified for its functional correctness using simulation tools and then processed for rough estimation of performance requirements. Generally, the DSE is manual or semiautomatic method used to estimate series of design evaluations. The DSE helps in selection of most suitable implementation of a design by analyzing its performance metrics like speed, area, development cost, power, design reliability, and flexibility. The modern embedded applications such as data streaming and processing methods demand efficient DSE methodologies to design optimal performance systems for multimedia tasks like speech processing, image processing, music and video processing and compression. These kind of streaming and processing tasks need high speed processing resources to provide high throughput in tasks execution. Energy efficiency of an embedded system also an important parameter for portable devices which operates on power constrained battery environments. Therefore, development of balanced optimized architectures for real time tasks is very important in order to consume less energy and provide required performance i.e. throughput. Nowadays an architecture called System on Chip (SoC) is a hybrid system which has been integrated on a chip with effective processing elements required to design a complete product. The functional elements which need to be implemented on SoC should be in the form of intellectual property (IP) and these IP cores can be designed along with GPP, ASIC, memories, and customized logic of FPGA. For example, the Virtex II Pro FPGA device capable of accommodate maximum of four parallel Power PC processors in its reconfigurable fabric. The SoC architecture provide many novel advanced features like reconfigurability, multilevel operating voltage, variable frequency, differential operational states and resizable memory etc. These kind of flexible resources make the SoC as the architectures suitable for execution of multi factious complex embedded applications. However, design space exploration is very crucial techniques to provide optimal solutions for real life applications.

In [16], a system-level design space exploration has been developed to evaluate architectural space exploration and also that enable the designers towards resources exploration process. These design exploration methods are used to evaluate the system-level parameters like components utilization, data throughput, and communication architecture and then enable the designer to produce optimal design. In [17], the design space refers to various embedded hardware software system configurations that satisfy design requirements whereas the DSE refers to the process of

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

systematically searching a design space for near optimal solutions with respect to given optimization objectives. Wherein [17], the DSE have been carried out by considering abstraction layers and viewpoints of a system. The abstraction layers support systems to model artifacts on different granularity devices and the viewpoints provides information of various perspectives a system on devices while focusing on system functional or technical requirements. Various required viewpoints like functional viewpoint, logical viewpoint and technical viewpoint were noticed in the research [17]. In [18], a profiling tool known as MEMTRACE has been presented for exploring applications complexity.

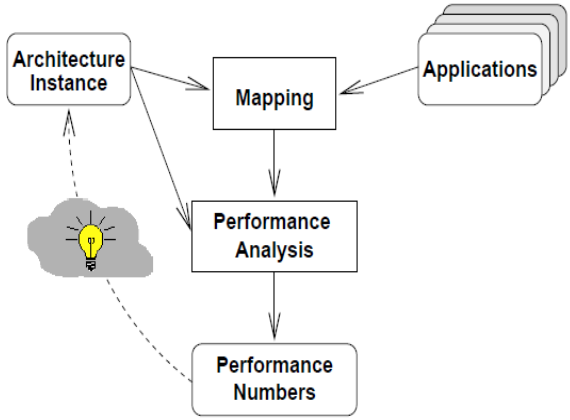

Figure 2: Y-Chart Approach For DSE [20]

Wherein [18], the software and hardware profiling methodologies has been examined and applied to the application H.264/AVC video decoder. A Behavioral Description Model (BDM) proposed in [19] presents various methodologies for design space exploration. A Y-chart approach has been presented [20] for Design Space Exploration and it is demonstrated in figure 2. The Y-chart DSE method enable the designer with quantitative design attributes captured by estimating the performance of the system architectures for given set of applications.

The Y-chart uses the following two steps:

1. Designers propose architecture and conduct performance analysis to construct an optimized model for the proposed architecture.
2. The performance analysis model can be evaluated for a set of mapped applications so that the designers can estimate optimal architecture for the mapped set of applications.

2.2 Hardware design space exploration

Hardware Software co-design can partition the specifications of an application into hardware and software specifications and demonstrate the behavior to satisfy the performance constraints. The Hardware Software Co-Design consists of two design levels are:

- *Hardware Software Partitioning:* It partitions the application specifications into hardware and software tasks specifications.
- *Scheduling:* It decides the execution order of the hardware and software tasks.

These hardware software partitioning and scheduling methods are interdependent and provide optimal solution. In the partitioning and scheduling process, the co-design algorithm considers hardware and software design parameters in order to take partitioning decisions. In general, there would be more than one design parameter exists for each task of an application. Therefore, the method of estimating design parameters or attributes of the tasks of an application for various computing platforms is called as hardware design space exploration.

3. PROPOSED METHODOLOGY

This section presents the research objective, On-chip heterogeneous architecture called Heterogeneous Reconfigurable Computing System (HRCS) and proposed DSE methodology for on-chip HRCS.

3.1 Research Objective

For a given application, resources optimization methodology has to be developed to utilize the resources of computing platform effectively while optimizing power and performance of the application.

This paper is intended to demonstrate a resources optimization methodology for Heterogeneous Computing Systems (HCS) in order to utilize the HCS resources effectively and also enhance the execution speed of the applications. The HCS is a computing platform consists of Programmable Logic Blocks (PLBs) array in combination with a General Purpose Processor (GPP) as Processing Elements (PEs).

3.2 On-chip Heterogeneous Computing Architecture

The heterogeneous computing systems enhance the execution speed of an application by co-scheduling the tasks to the resources of Heterogeneous

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Computing System (HCS) and also it helps in achieving maximum utilization of resources. But in real time, applications normally vary in terms of computing complexity and hence there is a need of flexible heterogeneous computing platforms equipped with softcore processor and reconfigurable hardware.

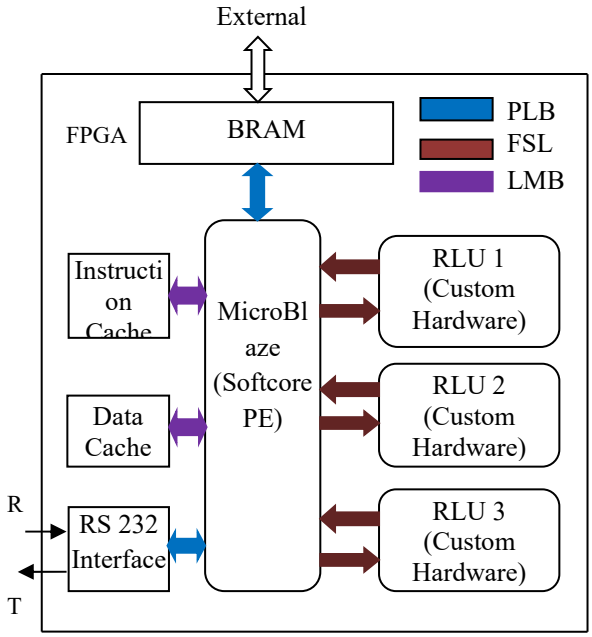

Figure 3: Heterogeneous Reconfigurable Computing System

In this research, HRCS architecture, shown in figure 3, has been realized on a single chip Virtex-5 FPGA device (XC5VLX110T) using Xilinx Embedded Development Kit (EDK). In the HRCS, a MicroBlaze soft core processor is configured in part of the reconfigurable area of FPGA and the rest reconfigurable area is used for configure multiple RLUs, memory and communication protocols. Wherein HRCS, the MicroBlaze is a 32-bit RISC architecture equipped with instruction and data cache memory of size 4KB each, for storing instructions as well as data while task execution. The RLU reconfigures its custom hardware for hardware tasks and also it support hardware task interface with off chip peripherals.

The on chip 64KB BRAM acts as shared memory for MicroBlaze and RLUs to store executable files, input and output data. BRAM memory controller, configured along with BRAM, helps in loading the task executable files, input and output data from external environment and also control inter process communication between BRAM and MicroBlaze. The data interchange between BRAM and custom hardware of RLU can happen through Fast Simplex Link (FSL) communication protocol. These on-chip functional blocks, MicroBlaze, RLUs, BRAM, instruction and data cache memory, are interconnected through 32-bit communication protocols like Processor Local Bus (PLB), Local Memory Bus (LMB) and FSL. The PLB provide interface between Soft core processor i.e. MicroBlaze and BRAM through BRAM controller that load instructions, input data and store back output data after computation. The communication protocol LMB supports interfacing of cache memories with MicroBlaze in orderto minimize memory access overheads. The FSL used to interface custom hardware configured in RLU with MicroBlaze and it has 32-bit FIFO implemented on BRAM to support data streaming between MicroBlaze and custom hardware.

3.3 Resources Estimation Methodology for HRCS

In order to achieve high efficiency in resource utilization, a Dynamic Resource Estimation Methodology (DREM) has been proposed for HRCS to estimate the design parameters and resources required for the tasks of an application. The DREM is shown in Figure 4 accepts the tasks of an application as input and executes them on the resources of computing platform HRCS to estimate design parameters of the application.

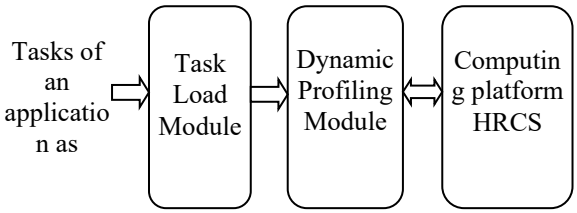

Figure 4: Dynamic Resource Estimation Methodology

The DREM has three sequential Modules called Task Load Module (TLM), Dynamic Profiling Module (DPM) and computing platform HRCS as shown in figure 4.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

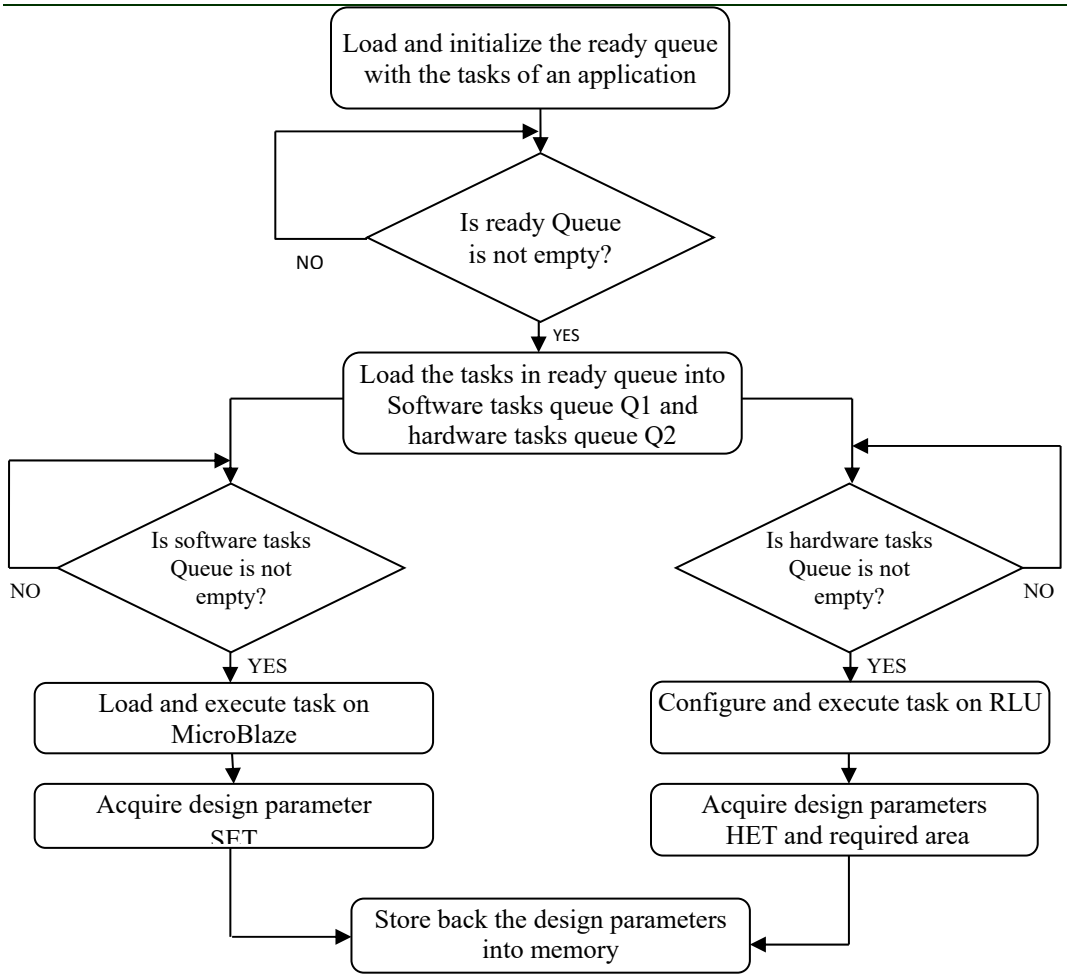

Figure 5: Flow Chart Of Dynamic Profiling Module For HRCS

The TLM provide interface between HRCS and I/O devices to accept the application as task graph. The DPM configure and executes the tasks of an application on the resources of HRCS to estimate the hardware software parameters of the tasks of an application. The behavior of DPM is described in the form of an algorithm as shown in figure 5. The DPM accepts the tasks which come across and insert them into software tasks queue Q1 and hardware tasks queue Q2. The tasks in Q1 execute only on soft core processor i.e. MicroBlaze configured on FPGA device and tasks in Q2 execute on RLU of FPGA device. The DPM verifies the availability of tasks in Q1 and Q2 before executing them on the resources of HRCS and acquire the following parameters.

- Software Execution Time(SET): Execution time of the task on soft core processor i.e. MicroBlaze
- Hardware execution Time(HET): Execution time of the task on FPGA
- Area required: Number of bit slices required for the tasks to implement on RLU of FPGA

The tasks binary code required for execution of the tasks on MicroBlaze and the tasks bit stream file required for execution of the tasks on RLU of FPGA. The binary code and bit stream file of the tasks along with their acquired parameters i.e. SET, HET and area required are stored in BRAM of HRCS for their future execution on the resources of HRCS.

4. EXPERIMENTAL RESULTS AND DISCUSSIONS

This section presents the FPGA resources utilization for construction of HRCS and the acquired hardware software design attributes for few DSP applications.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4.1 FPGA Resources Utilization for HRCS

In the targeted HRCS, MicroBlaze is equipped with instruction cache and data cache memories for storing program as well as data during executing the tasks of an application. The BRAM memory acts as shared memory for MicroBlaze and RLUs for storing input and output data. These functional blocks, MicroBlaze, RLUs (custom hardware for tasks of application), BRAM, Cache memory and general purpose I/O devices, have been interconnected through on-chip communication protocols like PLB, LMB and FSL as shown in figure 3. The architecture of HRCS has been configured on Virtex-5 FPGA device and the FPGA resources utilization for realization of computing platform HRCS is summarized in table 1.

Table 1: Virtex-5 FPGA (XC5VLX110T) Device utilization for HRCS

Resources	Module	Number of bit slices
Soft core (MicroBlaze)	MicroBlaze	1599
Hard core (Reconfigurable Logic Unit)	RLU1	500
	RLU2	500
	RLU3	500
Memory	DDR2 SDRAM (256MB)	1687
	dlmb cntlr (8KB)	7
	Ilmb cntlr (8KB)	4
Communication Interfaces (Bus controllers)	Dlmb	1
	Ilmb	1
	Mb plb	96
Debug Module	Mdm (64KB)	97
Timing and reset circuits	proc sys reset	30
	xps timer (64KB)	187
I/O interfaces	RS232 uart (64KB)	97
	DIP Switches 8bit (64KB)	67
	LEDs 8bit (64KB)	71
Heterogeneous Reconfigurable Computing System		5,444

4.2 Evaluation of design attributes

The DSP techniques, such as Linear Convolution, Circular Convolution, FFT, IFFT, FIR Filter and OFDM transmitter, are taken as test bench to evaluate the proposed methodology. As stated in DPM and Xilinx embedded system design process, the behaviour of the tasks in the DSP applications are described in C++ and also in HDL in order to execute them on soft core and hardcore processing elements of HRCS to acquire the hardware software design attributes. The C++ code of the tasks have been cross compiled by targeting to the soft core processor i.e. MicroBlaze and executable files were generated. The executables of the tasks are stored in program memory and then execution time of the tasks is acquired by executing them on MicroBlaze which is configured on FPGA device. Similarly, HDL code of the tasks was synthesized by targeting to Xilinx Virtex-5 (XC5VLX110T) FPGA devices and then gate level netlist has been generated. The gate level netlists are then mapped onto reconfigurable area of FPGA to generate configuration file required for task execution on FPGA. These configuration files are stored in memory and then the execution time and area required have been acquired by executing them on RLU of HRCS which is configured FPGA device. The tasks of DSP techniques were also implemented on popular general purpose processor Core 2 Duo [1.2GHz] and DSP processor TMS320C6713 DSK [225MHz]. The acquired hardware software attributes of the tasks of test bench applications are summarized in table 2.

An advanced application OFDM Transmitter is considered for estimating the performance of their tasks on HRCS. The task graph of the OFDM Transmitter is shown in figure 6.

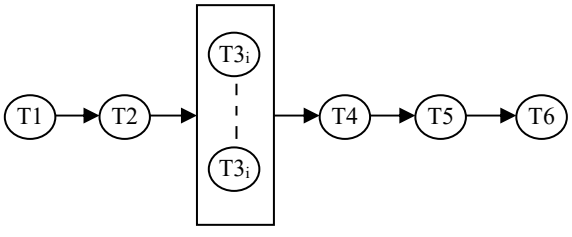

Figure 6: Task graph of OFDM Transmitter

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The tasks of OFDM transmitter are executed on MicroBlaze as well on RLU of HRCS and then the hardware software design attributes were acquired. The attributes of tasks of OFDM transmitter task graph are summarized in table 3.

Table 2: Hardware Software Design Attributes Of DSP Techniques

DSP Technique	Hardware Realization		Software Realization		
	Virtex-5 FPGA		MicroBlaze [125MHz]	Core 2 Duo [1.2GHz]	TMS320C6713 DSK [225MHz]
	Execution Time (ns)	Area(Number of slice)	execution Time (µs)	execution Time (µs)	execution Time (µs)
Linear Convolution	6.55	381	298.5	5	7.87
Circular Convolution	8.7	145	145	4	6.26
16-FFT	4.41	1704	28228	81.21	87.24
16-IFFT	5.349	225	26925	143.5	142.20
FIR Filter	19.67	536	3562	4	203.08
Binary generator	576	192	2191.7	500	1270
S/P conversion	1.216	13	192.4	300.5	8.012
16-QAM	6	16	128.5	4.5	0.864
P/S conversion	4.14	74	192.4	300.5	8.012
Cyclic Prefix	4.73	1917	382.3	6	8

Table 3: Hardware Software Design Attributes Of OFDM Transmitter

Tasks in Task Graph	Level of task	Hardware Realization		Software realization		
		Virtex-5 FPGA		MicroBlaze [125MHz]	Core 2 Duo [1.2GHz]	TMS320C6713 DSK 225MHz]
		Execution Time (ns)	Area (Number of slice)	Execution Time (µs)	Execution Time (µs)	Execution Time (µs)
Binary generator (T1)	1	576	192	2191.7	500	1270
S/P conversion(T2)	2	1.216	13	192.4	300.5	8.012
16-QAM(T3)	3	6	16	128.5	4.5	0.864
16-IFFT(T4)	4	8.646	334	444.0	143.5	142.20
P/S conversion(T5)	5	4.14	74	192.4	300.5	8.012
Cyclic Prefix (T6)	6	4.73	1917	382.3	6	8
OFDM Transmitter	7	600.73	2546	30012.3		

The application JPEG also considered as input to analyze the effectiveness of the heterogeneous computing platform HRCS. The JPEG is represented as task graph and their dependencies are shown in figure 7. The JPEG task graph consists of 7 tasks and the tasks behaviour is described as Gray conversion in T1, Matrix Transpose in T2, Wrapper 1 in T3, Wrapper 2 in T4, Quantization in T5, Encoding in T6, memory write back in T7.

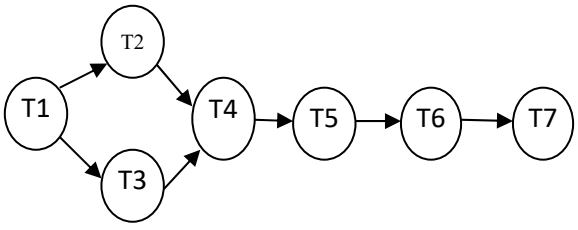

Figure 7: JPEG Task graph

The acquired design attributes of JPEG task graph is demonstrated in table 4. The table 4 presents execution time of the application JPEG on soft core rocessor and hardcore processing elements of HRCS. From the table 4, it is clear that the performance of the JPEG task graph is effective

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

when tasks are executed on reconfigurable area of FPGA as compared to soft core processors. The computation time of the application is more on MicroBlaze compared to Core 2 Duo processor. The number of CPU clock cycles taken by the application is equal to multiplication of clock speed of the processor and execution time of the application on the processor. So, the custom soft core MicroBlaze require more clock cycles compared to general purpose Core-2-duo soft core processor. Since the core 2 duo runs at 1.2 GHz, whereas MicroBlaze runs at 125MHz, the core 2 duo dissipates more power and also consumes more area. So, the MicroBlaze would be the soft core processor to design low power and performance effective computing platform for real time applications.

Table 4: Hardware Software Attributes Of JPEG Task Graph

Tasks in JPEG task graph	Level of the tasks	Hardware realization		Software realization	
		Execution time(ns)	Area (Number of slice)	(MicroBlaze) [125MHz] (ms)	Core 2 duo [1.2GHz] (µs)
Gray conversion (T1)	1	4.7	64	0.4	1.1
Matrix Transpose (T2)	2	4.372	64	0.4	1
Wrapper 1(T3)	2	6.081	128	6	10
Wrapper 2 (T4)	2	6..081	128	6	10
Quantization (T5)	3	5.259	64	2.6	3
Encoder (T6)	4	14.32	192	7.3	20
Memory Read/Write (T7)	5	4.372	64	0.4	1
JPEG		45.185	45.185	46.1	23.1

5. CONCLUSIONS AND FUTURE SCOPE

The modern embedded systems combine one or more processing cores to support dedicated logic running on an ASIC or FPGA and meet their design goals at optimized resource utilization. Since ever increasing system and application complexities, resources estimation become quite necessary to enhance the resources utilization for an application. So this paper presents a resource optimization methodology for on-chip heterogeneous computing systems.

5.1 Conclusion

This paper presented an on-chip multi resources computing system called Heterogeneous Reconfigurable Computing System (HRCS) modeled in Virtex-5 FPGA device and also a Dynamic Resource Estimation Methodology (DREM) is presented for the HRCS. A few DSP techniques were considered as test bench to demonstrate the behavior of DREM for the platform HRCS. The test bench consists of convolution, FFT, FIR filters, QAM, Binary generator, serial to parallel converter, parallel to serial converter, Cyclic Prefix, OFDM transmitter and JPEG encoder. The DSP techniques in test bench have been represented as task graphs and executed on HRCS computing resources, i.e. MicroBlaze soft core processor and RLU, and then the hardware software design attributes were acquired and tabulated. The tabulated hardware software design attributes, i.e. required area on FPGA, hardware execution time and software execution time, were stored in memory and they can be utilized for selecting the suitable computing resources for optimal performance of real life applications.

5.2 Future scope

This research was conducted to estimate the performance and resources requirement for an application. The resources estimation alone is not enough to design optimized computing system for real life applications. So the future research can be conducted to estimate design parameters like performance, memory usage, cache hit versus cache miss, energy consumption, etc. which also shows major impact in design of optimized computing systems for real life applications.

REFERENCES:

[1] Reiner Hartensstein, “Microprocessor is no more General Purpose: Why future reconfigurable platforms will win,” invited paper of the International conference on Innovative Systems in silicon.ISIS’97,

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Texas, USA, O-*ctober 8-10, 1997, pp 1-10.

[2] D. Wang, S. Li and Y. Dou, "Loop Kernel Pipelining Mapping onto Coarse-Grained Reconfigurable Architecture for Data-Intensive Applications," Journal of Software, Volume 4, no-1, p81-89, 2009.

[3] Mahendra Vucha and Lincy Sara Varghese, "Design Space Exploration of DSP Techniques for Hardware Software Co-Design: An OFDM Transmitter Case Study," International Journal of Computer Applications, Volume 116, Number 20, pp.29-33, April 2015.

[4] Mahmood Fazlali, Mojtaba Sabeghi, Ali Zakerolhosseini and Koen Bertels, "Efficient Task Scheduling for Runtime Reconfigurable Systems," Journal of Systems Architecture, Volume 56, Issue 11, pp. 623-632, November 2010.

[5] Yun Wang and Manas Saksena, "Scheduling Fixed –priority Tasks with Preemption Threshold," Sixth International Conference on Real-Time Computing Systems and Applications, RTCSA '99, 1999, pp 328-335.

[6] Ali Ahmadinia, Christophe Bodda and JurgenTeich, "A Dynamic Scheduling and Placement Algorithm for Reconfigurable Hardware," ARCS 2004, LNCS 2981, 2004, pp. 125 – 139.

[7] Xue-Gong Zhou, Ying Wang, Xun-Zhang Haung and Cheng-LianPeng, "On-line Scheduling of real time Tasks for Reconfigurable Computing System," International Conference on Computer Engineering and Technology, 2010, pp. 59-64.

[8] Zexin Pan and B. Earl Wells, "Hardware Supported Task Scheduling on Dynamically Reconfigurable SoC Architectures," IEEE Transactions on VLSI Systems, vol. 16, No. 11, pp.1465-1474, November 2008.

[9] HalukTopcuoglu, Salim Hariri and Min-You Wu, "Performance Effective and Low-Complexity Task Scheduling for Heterogeneous Computing," IEEE Transactions on Parallel and Distributed Systems, Vol. 13, No. 3, pp. 260 – 274, March 2002.

[10] S. Darba and D.P. Agarwal, "Optimal Scheduling Algorithm for Distributed Memory Machines," IEEE Trans. Parallel and Distributed Systems, Vol. 9, no. 1, pp. 87-95, Jan. 1998.

[11] Y.K. Kwok and I. Ahmad, "Dynamic Critical Path Scheduling: An effective Technique for Allocating Task Graphs to Multiprocessors," IEEE Trans. Parallel Distributed Systems, Vol. 7, no. 5, pp. 506-521, May 1996.

[12] S. Vassiliadis, S. Wong, G. N. Gaydadjiev, K.L.M Bertels, G. K. Kuzmanov and E. M. Panainte, "The Molen Polymorphic Processor," IEEE Transaction on Computers, Vol. 53, Issue 11, pp. 1363-1375, November 2004.

[13] Makimoto, T., "The hot decade of field programmable technologies," Proceedings of IEEE International Conference onField-Programmable Technology, 16-18 Dec. 2002, pp.3 – 6.

[14] Mouna Baklouti and Mohamed Abid, "Multi Soft-Core Architecture on FPGA," International Journal of Reconfigurable Computing, Volume 2014, Article ID 979327, 2014.

[15] Giovanni De Micheli and Rajesh K Gupta, "Hardware/Software Co-Design," Proceedings of the IEEE, Vol 85, No.3, March 1997

[16] Ramsey Hourani, Ravi Jenkal, W. Rhett Davis, Winser Alexander, "Automated Design Space Exploration for DSP Applications," Journal of Signal Processing Systems,Volume 56, Issue 2-3, September 2009, pp.199 – 216, Kluwer Academic Publishers Hingham, MA, USA

[17] Andy D. Pimentel, Member, CagkanErbas and Simon Polstra, "A Systematic Approach to Exploring Embedded System Architectures at Multiple abstraction Levels," 2006

[18] Matthias B¨uker, Stefan Henkler, Stefanie Schlegel, EikeThaden, "A Design Space Exploration Framework for Model-Based Software-intensive Embedded System Development," 2010

[19] Mahendra Vucha, Rajendra Patel and Arvind Rajawat, "Dynamic Profiling Methodology for Resource Optimization in Heterogeneous Computing System", Proceedings of International Conference on Emerging Research Computing, Information, Communication and Applications, ERCICA 2013, August 2013, pp. 41-46.

[20] Heiko Hubert and Benno Stabernack, "Profiling-Based Hardware/Software Co-Exploration for the design of Video Coding Architectures," IEEE Transactions on

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

circuits and systems for video technology, vol.19, No.11, November 2009.

[21] F.Thabet, P. Coussy, D. Heller, E. Martin, "Behavioral Description Model BDM for Design Space Exploration: A Case Study of HIS Algorithm for MC-CDMA System," 15th European Signal Processing Conference, 2007

[22] Bart Kienhuis, Ed F. Deprettere, Pieter van der Wolf, KeesVissers,"A Methodology to Design Programmable Embedded Systems: the Y- chart Approach,"LNCS series vol.2268, page 18-37, Springer-Verlag 2001.

[23] Jurgen Teich, "Hardware/Software Co-design: The Past, the present and predicting the future," Proceedings of the IEEE, 2012

[24] Karam S Chatha and Ranga Vemuri, "An Iterative Algorithm for Hardware-Software Partitioning, Hardware Design Space Exploration and Scheduling", Design Automation for Embedded Systems, 5, 281-293(2000)

[25] Mahendra Vucha and Arvind Rajawat, "Dynamic Task Distribution Model for On-Chip Reconfigurable High Speed Computing System," International Journal of Reconfigurable Computing, Volume 2015 (2015), Article ID 783237, 12 pages, December 2015.

[26] Michael C. McFarland, Alice C. Parker, Raul Carnposano, "Tutorial on High-Level Synthesis", Paper 23.1,25th ACM/IEEE Design Automation Conference, 1988 IEEE

[27] Naveen Kumar N, Rohit S., H Venkatesh Kumar, "FPGA Implementation of OFDM Transceiver using Verilog - Hardware Description Language", International Journal of Computer Applications, Vol 102, September 2014

[28] Mahendra Vucha and Arvind Rajawat, "A Novel Methodology for Task Distribution in Heterogeneous Reconfigurable Computing System," International Journal of Embedded Systems and Applications, Volume 5, Number 1, pp. 19-39, March 2015.

[29] Abhijit D. Palekar and Dr.Prashant V. Ingole, "OFDM System Using FFT and IFFT," IJARCSSE, Vol. 3, Issue 12, December 2013.

[30] Jonas StenbaekHegner, "Design of Power Efficient FPGA based Hardware Accelerators," DTU, IMM, 2012

[31] R. Durga Bhavani and D. Sudhakar, "Design and Implementation of Inverse Fast Fourier Transform for OFDM ", IJSEA, Vol. 2, Issue 7, 2013

[32] "TMS320C6713 DSK, Technical Reference", Spectrum Digital Inc, November 2003

[33] John G. Proakis, Dimitris G. Manolakis, "Digital Signal Processing," Pearson, Fourth Edition.

[34] http://www.xilinx.com/support/university/vivado/vivado-workshops/Vivado-adv-embedded-design-zynq.html.

[35] JernejBarbic, "Multi Core Architectures," Lecture notes, University of Southern California, Los Angeles, USA (www-bcf.usc.edu/~jbarbic/multi-core-15213-sp07.ppt).

[36] J. Jeon, D. Kim, D. Shin, and K Choi, Benchmark VHDL Examples, Design Automation Lab, http://dal.snu.ac.kr/index.php/Software/CDFG, Benchmark.tar.tz.

[37] MicroBlaze Processor Reference Guide, Embedded Development Kit (EDK 10.1i), http://www.xilinx.com/support/documentation/sw_manuals/mb_ref_guide.pdf.