IJIRSET
International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY

e-ISSN: 2319-8753 | p-ISSN: 2347-6710

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 10, Issue 12, December 2021

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 7.569

9940 572 462 | 6381 907 438 | ijirset@gmail.com | www.ijirset.com

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 12, December 2021 ||

| DOI:10.15680/IJIRSET.2021.1012177 |

A Survey on Operating System Design Challenges for Reconfigurable Computing

B.Abirami[1], V.Vasudevan[2]

Research Scholar, Department of Computer Science and Engineering, KARE, Tamilnadu, India[1]

Professor, Department of Computer Science and Engineering, KARE, Tamilnadu, India[2]

ABSTRACT: One of the major challenges in the design and development of Reconfigurable computing systems is that the conventional system designers are not familiar with the complexities of the modifications required in the conventional operating systems to control these reprogrammable hardware based system. One of the possible solutions is to provide standardized interfaces that are supported by the underlying operating system. In this paper, a historical view of the development of reconfigurable computing systems and the support provided by the operating systems is presented. This article summarizes the common pattern observed in these systems and also suggests the future direction of research in the design of reconfigurable computing systems in future.

KEYWORDS: Reconfigurable Computing, Operating Systems, Reprogrammable hardware, FPGAs.

I. INTRODUCTION

With the ever increasing volume of data processing in every application, there is a growing demand for computing power in every application domain. Most of the real world applications require their tasks to be executed with minimum possible delay and it is not possible for computing systems with general purpose CPUs to meet this requirement. As a result, computing systems with Field Programmable Gate Arrays (FPGAs) and Graphical Processing Units (GPUs) as accelerators have received great attention [1]-[2].

FPGAs are made up of a number of programmable blocks called Configurable Logic Blocks (CLBs) complemented with a number of specialized functional blocks , such as complex arithmetic and DSP blocks and transceivers. This makes the FPGAs suitable for any digital application. Since most of the logic blocks operate in parallel there is an increased speedup for many applications. Some of the recent works on Reconfigurable computer architectures are given in [3]-[8].
The current FPGA systems lack standard abstraction layers and this limits the interaction with the operating systems. Hence a reconfigurable computing system with FPGAs as accelerators needs an an Operating system (OS) which adopts a modular FPGA development flow and allows each system component to be changed and can interact with well defined hardware and software layers. Also to maximise the utilisation, the OS should employ resource-elastic scheduling.

This survey paper has been organized with four sections; Section 2 gives a brief introduction to Reconfigurable computing for the benefit of beginners along with references. Section 3 discusses the challenges in the design of reconfigurable computing systems. Section 4 briefly presents the research work carried out in this domain for the past few decades which are published in the literature. Section 5 gives a conclusion and future directions.

II. RECONFIGURABLE COMPUTING

Reconfigurable computing systems allow a customizable computing architectures but faces a number of design challenges. Some of the Industry sponsored initiatives are given in [9-11]. Developing a software for a particular application is much faster than hardware design because the hardware synthesis is time consuming and needs more time when the process is iterative.Standardization of hardware interfaces and high level synthesis can overcome this design time issue [12]. Another approach is to provide standardized interfaces at the operating system level which can provide a unified view not only for the developers but also for the users [13]. The FPGAs could be deployed in a number of applications, mainly because of the integration of reconfigurable devices with operating systems.

Design challenges: An operating system should make the hardware transparent and provide a well definedinterface to

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 12, December 2021 ||

| DOI:10.15680/IJIRSET.2021.1012177 |

the programmer [14]. Abstraction and resource management are the two important tasks of an OS.

Abstraction: An operating system's most elementary abstraction is a process [13]. The thread allows the tasks to run in parallel on the underlying hardware. Communication channels exist between processes and threads to coordinate their execution.

Resource management: Another important task of the OS is Resource Management. The available resources such as processors, internal and external memories, and other peripheral devices have to be shared by the processes and threads and hence they have to be managed and controlled by the OS.The FPGAs are presented to the OS as processors, memories or i/o devices in order to provide the integration. One of the first papers to present these concepts is [15]. The FPGAs can be used as accelerators or as parallel processing elements or as an additional independent processing element with its own tasks also. Partitioning, application loading, memory management and scheduling are some of the key services of an OS to support the Reconfigurable computing systems. Some of these services are briefly presented below along with adequate references;

FPGA Partitioning: FPGAs consist of hundreds of programmable elements (CLBs), dedicated functional elements such as DSPs and block RAMs and a set of routing resources to connect them in any desired manner.Several CLBs along with routing resources form a reconfigurable module known as an island [18]. Grouping FPGA resources as islands, one dimensional slots (1D slots) and two dimensional grids (2D grids) are presented in [16 -25].

Application Loading/Abstraction.FPGAs are used as accelerators to improve the computational performance of the systems. They are treated like any other hardware with their own device device drivers. The reconfigurable modules can be defined by means of standard interfaces and libraries as given in HybridOS (2007) [26] and LEAP (2011) [27, 28]. Based on how the reconfigurable module interacts with the CPUs they are defined as hardware application or hardware task or hardware process (BORPH (2007) [19]).

Placement and Scheduling: In any computational system, the OS schedules the execution of software tasks. The OS preempts a running task when higher priority tasks need to be executed. In reconfigurable computing, the placement of modules in reconfigurable areas is scheduled by the OS. For island based architectures, based on the size of the module the reconfigurable areas are identified and scheduled. For slots/grids based architecture the hardware resource requirements vary based on each hardware application. These architectures are presented in [29 -34]. Like software processes, the hardware applications can also be implemented in single tasking, multitasking with cooperative or preemptive scheduling. As context switching is difficult to implement, most of the implementations prefer single-task approach. Various implementation approaches have presented in [35 - 41].

III. LITERATURE REVIEW AND SURVEY

In the past three decades a number of research articles have been published reporting the ideas and implementations of framework and OS extensions in RC domain. They can be grouped into initial design ideas and standardized interface design and OS implementations as given in [13]. We have also reviewed those publications in the same categories and have briefly presented them in the following pages.

3.1. Initial Designs and Standardized interfaces;

A number of OS design issues have been published in the late 90s. Brebner (1996) presented a prototype OS for Xilinx FPGA XC6200 [15]. In this paper he has presented FPGA as an additional Virtual hardware, using smaller FPGA hardware and discussed the issues involved in such a design. The reconfigurable areas are controlled by the OS and are named as Swappable Logic Unit (SLU).

Compton et al.(2000) [22, 42] proposed the ideas to partition FPGAs into 2D grids with relocation and configuration defragmentations to reduce the reconfiguration overheads and also different scheduling approaches. These techniques reduced the configuration overheads by a factor of 8 to 12 compared to the traditional configuration methods.

Diesseler al.published several articles on operating system design issues for managing the reconfigurable hardware [16, 17, 23, 43, 44]. In these articles, the FPGA design tasks such as Design capture, partitioning, placement, routing, swapping and scheduling have been briefly presented. Reconfigurable models for multi tasking FPGAs and the operating functions needed such as task sizing, task placement, location independence, swapping, caching and pre-loading of tasks have been discussed.

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 12, December 2021 ||

| DOI:10.15680/IJIRSET.2021.1012177 |

3.2. API Frameworks for FPGAs

In literature a number of research articles which present the hardware modules with well defined standard interfaces and APIs for the convenience of the software developers have been published. Some of them have been presented in the following pages.

3.2.1. HybridOS (2007). In this work, a prototype reconfigurable computing system with Power PC as CPU embedded in and FPGA has been developed with Linux OS[26]. A prototype hybrid processor / FPGA accelerator execution model has been presented. Two alternative models namely direct access model and indirect access models have been presented [45].Three standard desktop applications namely, compression, audio processing and JPEG compression have been implemented and tested and better performance has been reported.

3.2.2. LEAP (2011). Latency-insensitive Environment for Application Programming (LEAP) provides abstraction for FPGAs, I/Os and memory devices [27, 28]. It provides compilation infrastructure with memory and communication facilities and separates computation and communication facilities to make the model insensitive to latencies [46].

3.2.3. RIFFA (2012). Reusable Integration Framework For FPGA Accelerators (RIFFA) uses reusable modules with well defined interfaces for both the hardware and software. It is reported that this work was carried out at University of California (USA). [47 - 48]. RIFFA provides standardized API for the hardware application and software modules with good communication interfaces and synchronization of application APIs between the reconfigurable modules and software modules.

3.3. Operating System for RCS: In the past three or four decades a number of leading world Universities and research laboratories have done extensive research and published their work on operating system extensions for designing reconfigurable hardware based systems. They are presented in this section with a short summary and references. Most of them are real implementations and completely control both the hardware modules and software tasks.

3.3.1. OS4RS (2003). This research was done by Mignolet et al. [36] and Nollet et al. [49] at KU Leuven (Belgium). This research has used an existing real time operating system RTAI and extended it to implement the run time task management. It has divided the reconfigurable architecture into a number of islands and dynamically scheduled tasks to a CPU or a reconfigurable island. The run time management tasks such as 2- level scheduler, uniform communication scheme, and task relocation have been implemented.

3.3.2. HThreads (2005). The hybrid thread programming model has been presented in [50 - 53] by the researchers at the University of Kansas. It presents a hybrid thread programming model using a multi threaded RTOS kernel. This design allows the programs written in to to be compiled to a hardware thread. The operating system provides a standardised interface for accessing hardware applications. The hardware provides static threads and does not use partial reconfiguration mode. Memory mapped registers are used for communication. In this research systems management is identified as the major task and scheduling and hardware synchronization have also been implemented. The scheduling algorithms such as round robin, preemptive priority, and FIFP have been implemented in this OS [52].

3.3.3. BORPH (2007). BORPH was developed and presented as a dissertation [19] at University of California at Berkeley (USA). In this work an entire FPGA is considered as an island.

Like other research works, BORPH was built as an extension of Linux OS and the process models, memory and file handling concepts are similar to that of UNIX processes. A new file format for application loading is implemented (BOF: BORPH object file). The OS decides whether a task should be executed by CPU or FPGA. There is no provision to preempt or swap a hardware task which is under execution. OS system calls are used for hardware and software communication

3.3.4. ReconOS (2007). The first version of the was built on the widely used Real time operating system eCos [54, 55]. The next version was developed based on the Linux operating system. In ReconOS also the FPGA resources are used as island to implement various hardware tasks. The CPU and the reconfigurable islands are interconnected via a bus. Another version of ReconOS has used the reconfigurable resources as 2 dimensional grids [35]. In the operating system a delegate software thread is assigned to each hardware thread. In a recent publication [56], it is reported that ReconOS implements a preemptive multitasking scheduling mechanism. S. The OS has been evaluated [55] using a Image and video processing application

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 12, December 2021 ||

| DOI:10.15680/IJIRSET.2021.1012177 |

3.3.5. CAP-OS (2010). In this research work, a Multiprocessor System-on-Chip (SoC) is used for task mapping, run-time scheduling, and resource management [57 – 61]. It uses the reconfigurable resources as islands with interconnecting NoC and designed for real time applications. The scheduling algorithm has been designed to meet the demand os the time sensitive applications. The hardware tasks are not preemptive. The hardware modules and software routines are managed by the SoC. The system has been designed to run a task either in conventional CPU or in FPGA.

3.3.6. R3TOS (2010). R3TOS is developed for real time tasks with reliability as the main focus and the details are presented in [62, 63] The tasks are defined using parallel software programming syntax and some of the tasks are designed to be executed in reconfigurable hardware. The reconfigurable resources are treated as two dimensional grid architecture for interconnection and communication purposes. Defective hardware regions are eliminated by rearranging the reconfigurable modules. It uses a non preemptive EDF scheduling policy to schedule the hardware tasks. The JPEG-compression application is used to evaluate the operating system.

3.3.7. FUSE (2011). Front-end USEr framework (FUSE) has presented an API conforming to the Portable Operating System Interface (POSIX) standard and has been integrated with PetaLinux [53]. An island style architecture is followed and the islands are connected to the system bus. In this work, a framework has been developed to provide the abstraction of hardware tasks. For each reconfigurable island a loadable software kernel module with well defined interfaces is provided for smooth handling of tasks. JPEGcompression, encryption and image processing applications have been used to evaluate the implementation.

3.3.8. SPREAD (2012). SPREAD uses the reconfigurable modules as a collection of islands with well defined standard interfaces [41, 65]. The system is designed for streaming applications with a well-defined interface for control and communication. PThreads model is used with software threads for scheduling the hardware tasks by the operating system. It allows hardware context switching and during run time allows a thread to switch between its hardware and software implementations. Cryptographic algorithms were used to evaluate the implementation and compared with BORPH,FUSE and ReconOS.

3.3.9. RTSM (2015). Run-Time System Manager (RTSM) employs the scheduling algorithms to select either a hardware module or a software version of the hardware module to execute a task.[37]. The tasks which are to be implemented in hardware are designed and stored as a configuration bit stream. In RTSM the size of the reconfigurable modules are different in size are connected to the system bus. RTSM employs context switching techniques to relocate hardware tasks among the reconfigurable areas. It was tested using an edge detection algorithm implementation

3.4. Discussion. The presented systems use island style architecture or treat the reconfigurable modules arranged as two dimensional grids The hardware applications are interconnected either through bus architecture or as Network on Chip or system bus structure. A P2P connection scheme is used in BORPH since it uses each FPGA as a separate island. RTSM uses a bus for controlling the hardware but also provides communication channels for streaming applications.

Except BORPH,all other implementations use POSIX thread for abstraction and fora task abstraction for time sensitive applications (eg..OS4RS). For hardware tasks, BORPH uses process abstraction model. SPREAD, RTSM, and ReconOS implemented Preemptive multitasking .Cooperative multitasking is considered in OS4RS

In most of the publications, the resource management task has been discussed theoretically and experimentally by simulation since there are limitations in partitioning, exchanging or preempting the resources in island style architecture. .

There seems to be no standardized benchmarking available to test and compare the Reconfigurable Computing systems except for image processing , cryptography and data compression applications.
Data security is another important area where the security issues were not discussed in detail.
For faster development and portability of applications, standardization is an important issue which should be addressed by the industry.

IV. CONCLUSION

In this work, a survey on the operating system tasks required to support the reconfigurable systems has been presented. The discussion gives the common patterns identified in these presentations. In most of the presentations, the hardware

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 12, December 2021 ||

| DOI:10.15680/IJIRSET.2021.1012177 |

task abstractions are implemented as POSIX Thread-based models ;The operating system assigns a delegate software thread for each hardware thread. The recent publications explore preemptive multitasking based scheduling for the hardware tasks. Benchmarks are available for image processing , cryptography and data compression applications. Dynamic partial reconfiguration and security aspects such as configuration and data security are the new research areas which can extend the scope of reconfigurable computing systems in future.

REFERENCES

[1] Z. Zhu et al., "A Hardware and Software Task-Scheduling Framework Based on CPU+FPGA Heterogeneous Architecture in Edge Computing," in IEEE Access, vol. 7, pp. 148975-148988, 2019, doi: 10.1109/ACCESS.2019.2943179.

[2] Alwarafy, K. A. Al-Thelaya, M. Abdallah, J. Schneider and M. Hamdi, "A Survey on Security and Privacy Issues in Edge Computing-Assisted Internet of Things," in IEEE Internet of Things Journal, doi: 10.1109/JIOT.2020.3015432

[3]A. Podobas, K. Sano and S. Matsuoka, "A Survey on Coarse-Grained Reconfigurable Architectures From a Performance Perspective," in IEEE Access, vol. 8, pp. 146719-146743, 2020, doi: 10.1109/ACCESS.2020.3012084.

[4] S. Tamimi, Z. Ebrahimi, B. Khaleghi and H. Asadi, "An Efficient SRAM- Based Reconfigurable Architecture for Embedded Processors," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 38, no. 3, pp. 466-479, March 2019, doi: 10.1109/TCAD.2018.2812118.

[5] H. Tang, D. Li, J. Wan, M. Imran and M. Shoaib, "A Reconfigurable Method for Intelligent Manufacturing Based on Industrial Cloud and Edge Intelligence," in IEEE Internet of Things Journal, vol. 7, no. 5, pp. 4248-4259, May 2020, doi: 10.1109/JIOT.2019.2950048

[6] M. P. Véstias, R. P. Duarte, J. T. De Sousa and H. C. Neto, "A Configurable Architecture for Running Hybrid Convolutional Neural Networks in Low-Density FPGAs," in IEEE Access, vol. 8, pp. 107229-107243, 2020, doi: 10.1109/ACCESS.2020.3000444.

[7] A. Kamaleldin, S. Hesham and D. Göhringer, "Towards a Modular RISC-V Based Many-Core Architecture for FPGA Accelerators," in IEEE Access, vol. 8, pp. 148812-148826, 2020, doi: 10.1109/ACCESS.2020.3015706.

[8] A. Vaishnav et al. "A survey on FPGA virtualization". International Conference on Field Programmable Logic and Applications (FPL), 2018.

[9] http://www.ccixconsortium.com/.

[10] http://www.hsafoundation.com/.

[11] Marcel Eckert, Dominik Meyer, Jan Haase, and Bernd Klauer, Operating System Concepts for Reconfigurable Computing: Review and Survey, International Journal of Reconfigurable Computing, 2016.

[12] J. Angermeier and J. Teich, "Heuristics for scheduling reconfigurable devices with consideration of reconfiguration overheads," in Proceedings of the 22nd IEEE International Parallel and Distributed Processing Symposium (IPDPS '08), pp. 1–8, April 2008.

[13] Marcel Eckert, Dominik Meyer, Jan Haase, Bernd Klauer, "Operating System Concepts for Reconfigurable Computing: Review and Survey", International Journal of Reconfigurable Computing, vol. 2016.

[14] A. S. Tanenbaum and H. Bos, Modern Operating Systems, Prentice Hall Press, Englewood Cliffs, NJ, USA, 2014.

[15] G. Brebner, "A virtual hardware operating system for the xilinx xc6200," in Field- Programmable Logic Smart Applications, New Paradigms and Compilers, R. Hartenstein andM. Glesner, Eds., vol. 1142 of Lecture Notes in Computer Science, pp. 327–336, Springer, Berlin, Germany, 1996.

[16] G. Wigley and D. Kearney, "The development of an operating system for reconfigurable computing," in Proceedings of the 9th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM '01), pp. 249–250, IEEE, Rohnert Park, Calif, USA, March 2001.

[17] G. B. Wigley and D. A. Kearney, "Research issues in operating systems for reconfigurable computing," in Proceedings of the International Conference on Engineering of Reconfigurable System and Algorithms (ERSA '02), 2002.

[18] D. Koch, C. Beckhoff, and J. Teich, "Recobus-builder—a novel tool and technique to build statically and dynamically reconfigurable systems for FPGAs," in *Proceedings of the International Conference on Field Programmable Logic and Applications (FPL'08)*, pp. 119-124, IEEE, Heidelberg, Germany, September 2008124, IEEE, Heidelberg, Germany, September 2008

[19] H. So and B. University of California, BORPH: An Operating System for FPGA-Based Reconfigurable Computers, University of California, Berkeley, Calif, USA, 2007, https://books.google.de/books?id=A3jVa48owLgC.

[20] J. Angermeier and J. Teich, "Heuristics for scheduling reconfigurable devices with consideration of

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 12, December 2021 ||

| DOI:10.15680/IJIRSET.2021.1012177 |

reconfiguration overheads," in Proceedings of the 22nd IEEE International Parallel and Distributed Processing Symposium (IPDPS '08), pp. 1–8, April 2008.

[21] O. Diessel, H. Eigindy, M. Middendorf, H. Schmeck, and B. Schmidt, "Dynamic scheduling of tasks on partially reconfigurable FPGAs," IEE Proceedings: Computers and Digital Techniques, vol. 147, no. 3, pp. 181–188, 2000

[22] A. Oetken, S. Wildermann, J. Teich, and D. Koch, "A busbasedSoC architecture for flexible module placement on reconfigurable FPGAs," in Proceedings of the 20th International Conference on Field Programmable Logic and Applications (FPL '10), pp. 234–239, Milano, Italy, September 2010.

[23] K. Compton, J. Cooley, S. Knol, and S. Hauck, "Configuration relocation and defragmentation for fpgas," Tech. Rep., 2000.

[24] F. Redaelli, M. D. Santambrogio, and S. OgrenciMemik, "An ILP formulation for the task graph scheduling problem tailored to bi-dimensional reconfigurable architectures," in Proceedings of the International Conference on Reconfigurable Computing and FPGAs (ReConFig '08), pp. 97–102, Cancun, Mexico, December 2008.

[25] K. Jozwik, H. Tomiyama, M. Edahiro, S. Honda, and H. Takada, "Rainbow: an OS extension for hardware multitasking on dynamically partially reconfigurable FPGAs," in Proceedings of the International Conference on Reconfigurable Computing and FPGAs (ReConFig '11), pp. 416–421, December 2011.

[26] J. Kelm, I. Gelado, K. Hwang et al., "Operating system interfaces: bridging the gap between cpu and fpga accelerators," in Proceedings of the International Symposium on FPGAs, vol. 48, Monterey, Calif, USA, February 2007.

[27] M. Adler, K. E. Fleming, A. Parashar, M. Pellauer, and J. Emer, "Leap scratchpads: automatic memory and cache management for reconfigurable logic," in Proceedings of the 19th ACM/SIGDA International Symposium on Field Programmable Gate Arrays (FPGA '11), pp. 25–28, ACM, Monterey, Calif, USA, March 2011.

[28] K. Fleming, H.-J. Yang, M. Adler, and J. Emer, "The LEAP FPGA operating system," in Proceedings of the 24th International Conference on Field Programmable Logic and Applications (FPL '14), pp. 1–8, Munich, Germany, September 2014.

[29] K. Bazargan, R. Kastner, and M. Sarrafzadeh, "Fast template placement for reconfigurable computing systems," IEEE Design and Test of Computers, vol. 17, no. 1, pp. 68–83, 2000.

[30] J. Teich, S. P. Fekete, and J. Schepers, "Optimization of dynamic hardware reconfigurations," The Journal of Supercomputing, vol. 19, no. 1, pp. 57–75, 2001.

[31] C. Steiger, H. Walder, and M. Platzner, "Operating systems for reconfigurable embedded platforms: online scheduling of realtime tasks," IEEE Transactions on Computers, vol. 53, no. 11, pp. 1393–1407, 2004.

[32]R. PellizzoniandM.Caccamo, "Real-time management of hardware and software tasks for FPGA-based embedded systems," IEEE Transactions on Computers, vol. 56, no. 12, pp. 1666–1680, 2007.

[33] D. Koch, C. Beckhoff, and J. Teich, "Minimizing internal fragmentation by fine-grained two-dimensional module placement for runtime reconfigurable systems," in Proceedings of the IEEE Symposium on Field Programmable Custom Computing Machines (FCCM '09), pp. 251–254, April 2009.

[34] A. Ahmadinia, J. Angermeier, S. P. Fekete et al., "ReCoNodesoptimization methods for module scheduling and placement on reconfigurable hardware devices," in Dynamically Reconfigurable Systems, pp. 199–222, Springer, Berlin, Germany, 2010.

[35] A. Wold, A. Agne, and J. Torresen, "Relocatable hardware threads in run-time reconfigurable systems," in Reconfigurable Computing: Architectures, Tools, and Applications: 10th International Symposium, ARC 2014, Vilamoura, Portugal, April 14–16, 2014. Proceedings, D. Goehringer,M. Santambrogio, J. Cardoso, and K. Bertels, Eds., vol. 8405 of Lecture Notes in Computer Science, pp. 61–72, Springer, Berlin, Germany, 2014.

[36] J.-Y. Mignolet, V. Nollet, P. Coene, D. Verkest, S. Vernalde, and R. Lauwereins, "Infrastructure for design and management of relocatable tasks in a heterogeneous reconfigurable system on- chip," in Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE '03), pp. 986–991, Munich, Germany, March 2003.

[37G. Charitopoulos, I. Koidis, K. Papadimitriou, and D. Pnevmatikatos, "Hardware task scheduling for partially reconfigurable fpgas," in Applied Reconfigurable Computing, K. Sano, D. Soudris, M. Huebner, and P. C. Diniz, Eds., vol. 9040 of Lecture Notes in Computer Science, pp. 487–498, Springer, Berlin, Germany, 2015.

[38] K. Rupnow,W. Fu, and K. Compton, "Block, drop or roll(back): alternative preemption methods for RH multi-tasking," in Proceedings of the IEEE Symposium on Field Programmable Custom Computing Machines (FCCM '09), pp. 63–70, April 2009.

[39] K.-J. Shih, H.-Y.Sun, and P.-A.Hsiung, "Dynamic hardware software task switching and relocation mechanisms for reconfigurable systems," in Proceedings of the IET International Conference on Frontier Computing. Theory, Technologies and Applications, pp. 157–162, Taichung, Taiwan, August 2010.

[40] P. G. Zaykov, G. K. Kuzmanov, and G. N. Gaydadjiev, "Reconfigurable multithreading architectures: a survey," in Embedded Computer Systems: Architectures,Modeling, and Simulation: 9th International Workshop, SAMOS

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 12, December 2021 ||

| DOI:10.15680/IJIRSET.2021.1012177 |

2009, Samos, Greece, July 20– 23, 2009. Proceedings, vol. 5657 of Lecture Notes in Computer Science, pp. 263–274, Springer, Berlin, Germany, 2009.

[41] Y. Wang, X. Zhou, L. Wang et al., "SPREAD: a streaming based partially reconfigurable architecture and programming model," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 21, no. 12, pp. 2179–2192, 2013.

[42] K. Compton, Z. Li, J. Cooley, S. Knol, and S. Hauck, "Configuration relocation and defragmentation for run-time reconfigurable computing," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 10, no. 3, pp. 209–220, 2002.

[43] O. Diessel and G. Wigley, "Opportunities for operating systems research in reconfigurable computing," Tech. Rep. ACRC-99- 018, University of South Australia, Adelaide, Australia, 1999.

[44] G. Wigley and D. Kearney, "The first real operating system for reconfigurable computers," in Proceedings of the 6th Australasian Computer Systems Architecture Conference (ACSAC'01), pp. 130–137, January 2001.

[45] J. H. Kelm and S. S. Lumetta, "HybridOS: runtime support for reconfigurable accelerators," in Proceedings of the 16th International ACM/SIGDA Symposium on Field Programmable Gate Arrays (FPGA '08), pp. 212–221, ACM, Monterey, Calif, USA, February 2008.

[46] L. P. Carloni, K. L. McMillan, and A. L. Sangiovanni-Vincentelli, "Theory of latency-insensitive design," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 20, no. 9, pp. 1059–1076, 2001.

[47] M. Jacobsen, Y. Freund, and R. Kastner, "RIFFA: a reusable integration framework for FPGA accelerators,"in Proceedings of the 20th IEEE International Symposium on Field-Programmable Custom Computing Machines (FCCM '12), pp. 216–219, IEEE, Toronto, Canada, April-May 2012.

[48] M. Jacobsen and R. Kastner, "RIFFA 2.0: a reusable integration framework for FPGA accelerators," in Proceedings of the 23rd International Conference on Field Programmable Logic and Applications (FPL '13), pp. 1–8, Porto, Portugal, September 2013

[49] V. Nollet, P. Coene, D. Verkest, S. Vernalde, and R. Lauwereins, "Designing an operating system for a heterogeneous recon figurableSoC," in Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS '03), p. 7, IEEE, April 2003.

[50] D. Andrews, D. Niehaus, R. Jidin et al., "Programming models for hybrid FPGA-CPU computational components: a missing link," IEEE Micro, vol. 24, no. 4, pp. 42–53, 2004.

[51] D. Andrews, D. Niehaus, and P. Ashenden, "Programming models for hybrid CPU/FPGA chips," Computer, vol. 37, no. 1, pp. 118–120, 2004.

[52] D. Andrews, W. Peck, J. Agron et al., "hthreads: a hardware/software co-designed multithreaded RTOS kernel," in Proceedings of the 10th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA '05), pp. 331–338, September 2005.

[53] W. Peck, E. Anderson, J. Agron, J. Stevens, F. Baijot, and D. Andrews, "Hthreads: a computational model for reconfigurable devices," in Proceedings of the International Conference on Field Programmable Logic and Applications (FPL '06), pp. 885–888, Madrid, Spain, August 2006.

[54] E. L¨ubbers and M. Platzner, "Reconos: an RTOS supporting hard- and software threads," in Proceedings of the International Conference on Field Programmable Logic and Applications (FPL '07), pp. 441–446, IEEE, Amsterdam, The Netherlands, August 2007.

[55] A. Agne, M. Happe, A. Keller et al., "ReconOS: an operating system approach for reconfigurable computing," IEEE Micro, vol. 34, no. 1, pp. 60–71, 2014.

[56] M. Happe, A. Traber, and A. Keller, "Preemptive hardware multitasking in reconOS," in Applied Reconfigurable Computing: 11th International Symposium, ARC 2015, Bochum, Germany, April 13–17, 2015, Proceedings, vol. 9040 of Lecture Notes in Computer Science, pp. 79–90, Springer, Berlin, Germany, 2015.

[57] D. Gohringer and J. Becker, "High performance reconfigurable multi-processor based computing on FPGAs," in Proceedings of the IEEE International Symposium on Processing, Workshops and Phd Forum (IPDPSW '10), pp. 1–4, Atlanta, Ga, USA, 2010.

[58] D. G¨ohringer, M. H¨ubner, L. Hugot-Derville, and J. Becker, "Message passing interface support for the runtime adaptive multi-processor system-on-chip RAMPSoC," in Proceedings of the 10th International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation (IC-SAMOS '10), pp. 357–364, July 2010.

[59] D. G¨ohringer, M. H¨ubner, E. N. Zeutebouo, and J. Becker, "CAP-OS: operating system for runtime scheduling, task mapping and resource management on reconfigurable multiprocessor architectures," in Proceedings of the IEEE International Symposium on Parallel and Distributed Processing, Workshops and PhD Forum (IPDPSW '10), pp. 1–8, April 2010.

[60] D. G¨ohringer, S. Werner, M. H¨ubner, and J. Becker, "RAMP- SoCVM: runtime support and hardware

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 12, December 2021 ||

| DOI:10.15680/IJIRSET.2021.1012177 |

virtualization for a runtime adaptive MPSoC," in Proceedings of the 21st International Conference on Field Programmable Logic and Applications (FPL '11), pp. 181–184, IEEE, Chania, Greece, September 2011.

[61] D. Gohringer, M. Hubner, V. Schatz, and J. Becker, "Runtime adaptive multi-processor system-on-chip: RAMPSoC," in Proceedings of the IEEE International Symposium on Parallel and Distributed Processing (IPDPS '08), pp. 1–7, Miami, Fla, USA, April 2008.

[62] X. Iturbe, K. Benkrid, A. T. Erdogan et al., "R3TOS: a reliable reconfigurable real-time operating system," in Proceedings of the NASA/ESA Conference on Adaptive Hardware and Systems (AHS '10), pp. 99–104, IEEE, Anaheim, Calif, USA, June 2010.

[63] X. Iturbe, K. Benkrid, C. Hong et al., "R3TOS: a novel reliable reconfigurable real-time operating system for highly adaptive, efficient, and dependable computing on FPGAs," IEEE Transactions on Computers, vol. 62, no. 8, pp. 1542–1556, 2013.

[64] A. Ismail and L. Shannon, "FUSE: front-end user framework for O/S abstraction of hardware accelerators," in Proceedings of the 19th IEEE International Symposium on Field-Programmable Custom ComputingMachines (FCCM'11), pp. 170–177,May 2011.

[65] Y. Wang, J. Yan, X. Zhou et al., "A partially reconfigurable architecture supporting hardware threads," in Proceedings of the International Conference on Field-Programmable Technology (FPT '12), pp. 269–276, IEEE, Seoul, Republic of Korea, December 2012.

INNO SPACE
SJIF Scientific Journal Impact Factor

Impact Factor:
7.569

doi cross ref

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

Scan to save the contact details