Current Trends in Technology and Sciences
ISSN: 2279- 0535. Volume: 3, Issue: 5 (Aug-Sept. 2014)

Reconfigurability in FPGA's

G.Prasad gurram_p@yahoo.com,
N.Vasantha vasantha54321@gmail.com,
C V Srinivas chalamchala_vas@yahoo.co.in,
B.Lakshmi lrsbayyalakshmi@gmail.com

Abstract - FPGAs keep getting larger and faster. They have reached a level where a whole 32 bit CPU fits into a single FPGA and doesn't even fill it. So FPGAs can house quite large logic circuits. Another development branch leads to dynamically reconfigurable FPGAs. That means that certain areas within the FPGA can be reconfigured while the rest continues to run unaffected. The next step is to combine these two abilities. In this paper we show how we implemented a on chip memory on an FPGA and combined it with an PCI –X IP core which can be in master read and master write mode of operation while the memory continues to run unaffected.
We explain the steps it took to build the memory on the FPGA, interface to a FIFO to buffer data to and from the FPGA, and to get two different operations to work. These operations are the memory read control and memory write control of the PCI core on the FPGA, they will be replaced on requirement with dynamic reconfiguration. Memory Read operation will be used for data read out from the hard disk through external FIFO, on chip memory and serializing, and Memory write operation will read data from the on chip memory, to external FIFO and through the PCI-X core to the cache and hard disk Satellite Data Acquisition.

Key words - FIFO, PCI IP Master Core, DMA, AHDL, CPLD

1. INTRODUCTION

Reconfigurable computing is the process of changing the structure of a reconfigurable device at start-up time respectively at run time. The logic block is defined by its internal structure and granularity. The structure defines the different kinds of logic that can be implemented in the block, while the granularity defines the maximum word length of the implemented functions. The functionality of the logic block is obtained by controlling the connectivity of some basic logic gates or by using LUTs and has a direct impact on the routing resources. As the functional capability increases, the amount of logic that can be packed into it increases. A collection of CLB/LAB known as a logic cluster, is described with the following four parameters: the size of (number of inputs to) a LUT, the number of CLB/LAB in a cluster, the number of inputs to the cluster for use as inputs by the LUT and the number of clock inputs to a cluster (for use by the register). Thus the size and complexity of the basic computing blocks is referred to as the block granularity. All the reconfigurable platforms based on their granularity are distinguished into two groups, the fine grain and coarse grain systems. A number of reconfigurable systems use a granularity of logic block that we categorize as medium grained. Medium grained logic blocks may be used to implement data path circuits of varying bit widths, similar to the fine grained structures. Fine grain array has many configuration points to perform very small computations and thus requires more data bits during configuration. This is because the basic programmed building block consists of a combinatorial network and a few flip-flops in fine grained architectures. The fine grain programmability is more amenable to control functions, while the coarser grain blocks with arithmetic capability are more useful for data path operations.

Dynamic Reconfiguration is the ability to update only a portion of the configuration memory in an FPGA with a new configuration without stopping the functionality of the unchanged section of the FPGA. Dynamic Reconfiguration enlarges the design space for developers. Different logic functions can be stored in memory until the need arises for them to be configured into the FPGA.

Recent advances in the manufacturing process promise 50 million gates of reconfigurable Logic at substantially lower costs. The increased gate count along with richer embedded feature sets has greatly improved the economics for using Reconfigurable Technology. One single FPGA can simultaneously carry various complex cores like processors, pseudo random generators, control logic and filters just to name a few of them. Dynamic Reconfiguration allows replacing a specific core if a new function is required. This situation is similar in the manner with computers with large hard drives storing applications for days before they are loaded into memory. Imagine a system which uses five different cores over the time, but not more than three simultaneously. Without dynamic reconfiguration you would either need a huge FPGA which can carry all cores at once, or you would need three individual FPGAs which will be fully reconfigured. The first case is a waste of FPGA area. The second case implies increased hardware costs and power consumption. Some cores support dual functions. With dynamic reconfiguration an FPGA which has the size to carry three cores (for all occurring combinations) will suffice. All cores are stored in memory. On requirement, an unused core can be replaced with a new core by a partial bit stream. The difference to a full reconfiguration is that the other cores aren't affected by the reconfiguration and keep their state.

Copyright © 2014 CTTS.IN, All right reserved

2.QUARTUS SOFTWARE AND ALTERA HARDWARE DESCRIPTIVE LANGUAGE

The Altera Quartus II design software is a multiplatform design environment that easily adapts to specific needs in all phases of FPGA and CPLD design. Quartus II software delivers the highest productivity and performance for Altera FPGAs, CPLDs, and Hard Copy ASICs. Quartus II software delivers superior synthesis and placement and routing, resulting in compilation time advantages. Compilation time reduction features include, Multiprocessor support, Rapid Recompile, Incremental compilation. Quartus II Analysis and Synthesis, together with the Quartus II Fitter, incrementally compiles only the parts of your design that change between compilations. By compiling only changed partitions, incremental compilation reduces compilation time by up to 70 percent. For small engineering change orders (ECOs), the Rapid Recompile feature maximizes your productivity by reducing your compilation time by 65 percent on average, and improves design timing preservation.

AHDL is a proprietary digital Hardware Description Language (HDL) from Altera Corporation for programming their Complex Programmable Logic Devices (CPLD) and Field Programmable Gate Arrays (FPGA). This language has an Ada programming language-like syntax and similar operation to VHDL or Verilog. It is supported by Altera's Quartus and Max+ series of compilers. An advantage of AHDL is that all language constructs are synthesizable. AHDL is to Verilog much as assembly language is to a higher-level programming language: in AHDL, you have more control.

3. FPGA 90NM STRATIX EP1S25F1020C5

The Stratix FPGA is used to implement the following modules. Static configuration of on chip memory and PCI core, interface & control logic, and Dynamic Configuration logic. Stratix devices contain a two-dimensional row- and column-based architecture to implement custom logic [19]. A series of column and row interconnects of varying length and speed provides signal interconnects between logic array blocks (LABs), memory block structures, and DSP blocks. The logic array consists of LABs, with 10 logic elements (LEs) in each LAB. An LE is a small unit of logic providing efficient implementation of user logic functions. LABs are grouped into rows and columns across the device.

M512 RAM blocks are simple dual-port memory blocks with 512 bits plus parity (576 bits). These blocks provide dedicated simple dual-port or single-port memory up to 18-bits wide at up to 318 MHz. M512 blocks are grouped into columns across the device in between certain LABs. M4K RAM blocks are true dual-port memory blocks with 4K bits plus parity (4,608 bits). These blocks provide dedicated true dual-port, simple dual-port, or single-port memory up to 36-bits wide at up to 291 MHz. These blocks are grouped into columns across the device in between certain LABs. M-RAM blocks are true dual-port memory blocks with 512K bits plus parity (589,824 bits). These blocks provide dedicated true dual-port, simple dual-port, or single-port memory up to 144-bits wide at up to 269 MHz. Several M-RAM blocks are located individually or in pairs within the device's logic array.

Digital signal processing (DSP) blocks can implement up to either eight full-precision 9 × 9-bit multipliers, four full-precision 18 × 18-bit multipliers, or one full-precision 36 × 36-bit multiplier with add or subtract features. These blocks also contain 18-bit input shift registers for digital signal processing applications, including FIR and infinite impulse response (IIR) filters. DSP blocks are grouped into two columns in each device. Each Stratix device I/O pin is fed by an I/O element (IOE) located at the end of LAB rows and columns around the periphery of the device. I/O pins support numerous single-ended and differential I/O standards. Each IOE contains a bidirectional I/O buffer and six registers for registering input, output, and output-enable signals. When used with dedicated clocks, these registers provide exceptional performance and interface support with external memory devices such as DDR SDRAM, FCRAM, ZBT, and QDR SRAM devices. High-speed serial interface channels support transfers at up to 840 Mbps using LVDS, LVPECL, 3.3-V PCML, or HyperTransport technology I/O standards.

4. STATIC CONFIGURATION OF ON CHIP MEMORY AND PCI IP MASTER CORE.

In the on chip implementation and PCI core mentioned below each module consists of one configuration. The distinctive feature of this static configuration is that it consists of single system wide configuration. Prior to commencing an operation, the reconfigurable resources are loaded with their respective configurations. Once operation commences, the reconfigurable resources will remain in this configuration throughout the operation of the application. Thus hardware resources remain static for the life of the design and static reconfiguration allocates logic for the duration of an application. In our design the on chip is designed to store 1K Qwords of memory during Master write or during Master read function. The PCI core can be statically reconfigured to function as a PCI Master read or PCI master write depending on the application. In both the situations the on chip memory is unaltered and only the application functions are reconfigured. Thus both the functions were realized in the single FPGA and also in the same PCB.

On-chip memory is the simplest type of memory for use in an FPGA-based embedded system. The memory is implemented in the FPGA itself; consequently, no external connections are necessary on the circuit board. On-chip memory is the highest throughput, lowest latency memory possible in an FPGA-based embedded system. It typically has a latency of only one clock cycle. Memory transactions can be pipelined, making a throughput of one transaction per clock cycle typical. Some variations of on-chip memory can be accessed in dual-port mode, with separate ports for read and write transactions. Dual-port mode effectively doubles the

Copyright © 2014 CTTS.IN, All right reserved

potential bandwidth of the memory, allowing the memory to be written over one port, while simultaneously being read over the second port. Another advantage of on-chip memory is that it requires no additional board space or circuit-board wiring because it is implemented on the FPGA directly. Using on chip memory can often save development time and cost. Finally, some variations of on-chip memory can be automatically initialized with custom content during FPGA configuration. This memory is useful for holding small bits of boot code or LUT data which needs to be present at reset.

The on chip memory was designed using the M RAM block of the Stratix device as shown in Table 1. In our design the FPGA memory was built to accommodate 1000 Q Words (64bit) and each frame is of 302 Q Words. The FPGA on chip memory was designed as two memory banks of capacity 1000 Q words each. Using the alternate buffer concept the read and write operations were designed so that 1000 words i.e. 3 frames of data can be written in one buffer and then the write operation will switch to the next buffer and remaining 1000Q Words will be written.

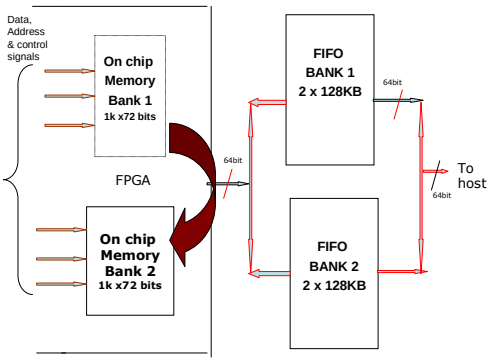

Figure 1. On chip interface to Onboard FIFO

On chip Memory Implementation using the LPM function of Altera Quartus

When the first bank writing is complete a control signal will be issued by the memory controller so that the first buffer data can be read on to the external FIFO, thus the 3 frames written in the first bank will be written to the external FIFO by enabling the write control signal and write clock of the external FIFO. After read out from the first on chip memory bank now this bank will be ready to take in the next 3 frames. Next the second memory bank will be writing to the external FIFO and this memory will also be available for next 3 frames. This will result in continuous write and read between on chip memory and external FIFOs. Thus a large memory of desired volume can be realized. Read operation of the external FIFO will empty the FIFO thus in this way required volume of data from multiple inputs can be written and stored.

Device	RAM Name	RAM (banks)	Size (bits)	Confi guration
Xilinx Virtex	Block RAM	8 - 208	4096	4096 x1 2048 x2
Altera Apex E	Embedded system block	12 -216	2048	2048 x1 1024 x 2
Stratix	M-RAM blocks	3 -138	589,824	8K × 72

Table 1. FPGA On chip RAMs

Total Memory bits that were required to realize the above design was 573456bits out of 1944576 bits (i.e. 29.49% of the memory bits) and since toggling between high frequency clock (read) and low frequency clock (write) a power saving of 10% was observed.

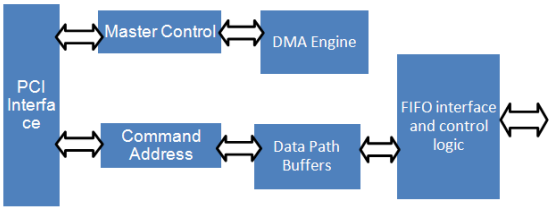

Figure 2. PCI CORE

The PCI core is a dual functionality core which enables to read and write to system memory.The FPGA is statically configured for PCI Master Read and PCI Master write operations. Depending on the requirement the corresponding configuration is loaded and will remain till the operation is complete. If change of functionality is required than again the required functionality is configured till the operations are complete.

5.PCI MASTER READ

To initiate a master read transaction, the master control logic requests the bus by asserting the lm_reqn signal to the local side of the pci_core function [16] for a 64-bit transaction. At the same time, the DMA engine signals the FIFO interface to request an FIFO access. The PCI function asserts reqn to request the PCI bus and simultaneously asserts lm_tsr from the local side to indicate that the master is requesting the PCI bus. On receiving the gntn signal from the arbiter, the pci_core function asserts the lm_adr_ackn and lm_tsr local side signals, indicating that the bus has been granted and the local side must supply the starting address and command. The master control provides the PCI address on l_adi and the PCI command on l_cbeni during the same clock cycle that the lm_adr_ackn and lm_tsr signals are asserted. The master control asserts the lm_rdyn input to the pci_core function to indicate that it is ready to accept data. The pci_core function asserts lm_ackn, indicating to the master control that it has registered data from the PCI side on the previous clock cycle and is ready to send data to the local side master interface. Because lm_rdyn was asserted in the previous clock cycle and lm_ackn is asserted in the current cycle, the function asserts lm_dxfrn to indicate a valid data transfer on the local side. The pci_mtcore function also asserts lm_tsr, indicating to the master control that a data phase was completed successfully on the PCI bus during the

Copyright © 2014 CTTS.IN, All right reserved

previous clock. The FIFO interface logic reads one WORD of data at a time from the PCI-to- FIFO buffer, writes into onchip memory and later shifts data into the external FIFO. During a master read transaction, the master control logic asserts the stop signal to the DMA and the FIFO interface if the master must release the bus prematurely. The lm_lastn signal is asserted to the core to end the transaction normally, i.e., all data has been received. When the lm_lastn signal is asserted, the core deasserts framen as soon as possible and asserts irdyn to read the last data on the PCI bus. Additionally, the FIFO interface transfers all valid data from the PCI-to-FIFO buffer and stops the operation. If the master has not read all of the data from the target due to premature termination, the master control logic and the DMA engine start a new master read cycle to read the remaining data. During master read the data from hard disk will be read out to the PCI FIFO, to on chip memory and than to external FIFO on the board. During master read transactions, the FIFO interface logic controls data from the PCI FIFO to the external FIFO buffer.

6. PCI MASTER WRITE

To initiate a master write transaction The DMA sends a signal to the FIFO interface to request an FIFO access. The master control logic waits for the external FIFO -to-PCI FIFO buffer to be filled with a predetermined number of WORDs before requesting the PCI bus. This action ensures that the master does not violate PCI latency protocol because of slow FIFO reads. If the transfer count is less than required WORDs, the master control logic waits for all of the data to be ready in the FIFO -to-PCI FIFO buffer before requesting the bus. If the transfer count is more than or equal to required WORDs, it waits for 32 WORDs to be written into the FIFO buffer before requesting the bus by asserting lm_reqn signal. On receiving gntn, the pci_core function asserts lm_tsr[1] and lm_adr_ackn. This action indicates to the local side that the bus has been granted and the local side must provide a valid address and command on l_adi and l_cbeni, respectively. The master control logic asserts lm_rdyn to the pci_core function to indicate that it is ready to transfer the data from external FIFO -to-PCI. During master write the data from on chip memory will be read out to the external FIFO, than -to-PCI FIFO in the core, to the hard disk through the cache memory. During master write transactions, the FIFO interface logic controls data from the external FIFO to the PCI FIFO buffer. Because of the reconfigurability feature the master read and master write transactions are implemented on the same board. By configuring the board for master write operations data is read from the external FIFO to the hard disk and by configuring for the master read operations date is read from the hard disk to external FIFO, without changing the total design only the interface logic needs to be changed to achieve this. The data path FIFO buffers provide an internal buffer for the data transferred between the PCI bus and the external FIFO. The PCI core has the two FIFO buffers each of 128 x 32 size, one will be used for transactions from PCI core to external FIFO and other from external FIFO to PCI Core depending on the selection of Master_write or Master_read.

7. DYNAMIC CONFIGURATION

The logic is designed to cater to past, present and future satellite missions. The frame formats of every satellite are different, with selectable configuration logic the desired satellite whose data is to be acquired is selected depending on the selection parameters. Applying the principle of Dynamic Reconfiguration [20] only the portion of configuration memory in an FPGA corresponding to the satellite selected is enabled without stopping the unchanged section of the FPGA. Different logic functions are stored in memory and dynamic allocation scheme re-allocates hardware at run time. By dynamically loading the parameters during operation, system performance increases. The functional configuration of the device is modified or changed during operation through the software via the PCI core or offline through a front panel switch. Here the physical hardware is much smaller than the sum of the resources required by each of the configurations. Therefore instead of reducing the number of configurations that are mapped, we swap them in and out of the actual hardware as and when they are needed. Using the reconfigurable facility, if one satellite is selected the parameters corresponding to that satellite get loaded and we will be able to acquire data from the selected satellite.

The on chip memory configuration is fixed and will support both PCI Master read and PCI Master write modes of operation. Thus using the principle of dynamic configuration only the PCI modes are reconfigured but the memory part of the configuration is unchanged. For different satellites the number of frames to fit into memory may be changed accordingly.

8. CONTROL AND INTERFACE LOGIC.

During PCI master write the interface control logic will check for the status of the on board FIFOs and when one bank shows half full status, it makes that bank available for read to the PCI core. Once the FIFO half full flag is set the read operation is initiated and data is read from fifo to host system through the PCI core. The write frequency is derived from the input clock and is slow. The read frequency of the on chip RAM is derived from the input clock and is of high frequency than the write clock. The same read clock will be used to write into the external FIFO shown in Table 2. Thus the reading and writing are done without conflicts and errors like data overflow, data overrun etc.

During PCI master read the interface control logic will monitor the status of the PCI core FIFO and depending on its status will readout the data to the on chip memory and to the external FIFO. This will use the PCI clock which is very fast for read out and write to on chip memory. The read out from the on chip memory and external FIFO is a slower clock compared to the PCI clock.

Copyright © 2014 CTTS.IN, All right reserved

The multi-context architecture includes multiple memory bits for each programming bit location. These memory bits can be thought of as multiple planes of configuration. Only one place of configuration information can be active at a given moment, but the architecture can quickly switch between different planes, or contexts, of already programmed configurations. Using this principle the control and interface logic will switch between PCI master write and PCI master read.

9. SIMULATION RESULT

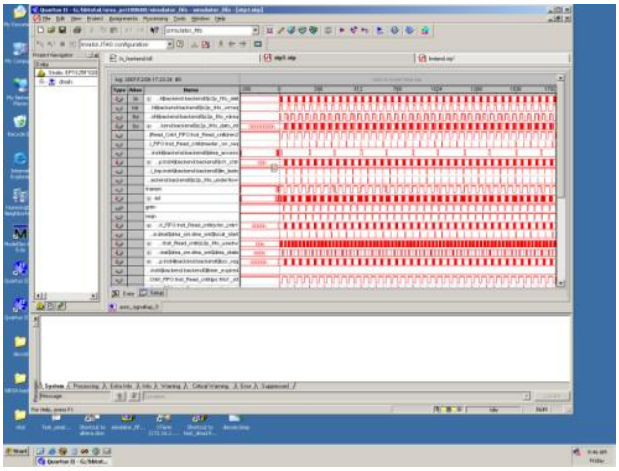

Figure 3 DMA burst in chaining mode.

10. CONCLUSION

The Reconfigurable computing enables to use a single board for PCI Master write or PCI Master read depending on the application for which it is intended to be used. Without Reconfigurable computing two different boards have to be designed and fabricated for testing out the two different applications. The FPGA based PCI core interface, and associated logic designed and developed is suitable for satellite data acquisition systems in the Ground segment. The throughput to the host achieved is between 100 to 220Mbytes/second thus it can cater to high speed data acquisition. Since the major modules are incorporated into the FPGA a power reduction of nearly 20% is achieved in the design. The logic is validated with an inbuilt simulator so that the total chain involved in the design is completely tested.

REFERENCES

[1] DDR and DDR2 SDRAM High-Performance Controllers and ALTMEMPHY IP User Guide section in volume 3 of the External Memory Interface Handbook

[2] On-Chip FIFO Memory Core in Volume 5: Embedded Peripherals of the Quartus II

[3] IDT FIFO Reference Guide

[4] Memory Interfaces made easy with Xilinx FPGAs and memory Interface Generator, WP260(v1.0) February 16,2007.

[5] Developing high speed memory interfaces. The Lattice SCM FPGA advantage White paper February 2006.

[6] Memory System Design from Altera Corporation February 2010.

[7] Global Memory Mapping for FPGA based Reconfigurable systems, Iyad Quaiss and Ranga Vemuri

[8] J. Toledo, H.Muller, J. Buytaert, F.Bal, A.David, A.Guirao and F.J.Mora (2002), "A plug and play approach to data acquisition", Network architecture and performance digital equipment corporation, Littleton

[9] Charles Geber, Kevin Yee (2000), "Peripheral component interfaces with quick logic QL1624 FPGA", quick logic corporation, Santa Clara.

[10] Jim McManus, PCI Applications Engineer, Xilinx Inc " Using FPGAs as a flexible PCI Interface Solution".

[11] P.Assis, P.Brogueira, L.Melo, M.Pimenta, J.c.Silva, J.Varela LIP-Lisbon, Portugal(2003), 28th International Cosmic Ray Conference. " A PCI based data acquisition system for Ground Array Detectors with Wireless Synchronization Through GPS".

[12] David Robinson, Patrick Lysaght, Gordon M^{c}Gregor and Hugh Dick "Performance Evaluation of a Full Speed PCI Initiator and Target Subsystem using FPGAs".

[13] Daniel Ziener, Jurgen Teich " Power Signature Watermarking of IP Cores for FPGAs.

[14] Haber .J (2003)." Using a commercial IP core in space flight avionics. Lessons learned".

[15] Pillem Ramesh, Venkata Aravind Bezawada, K S N Vittal, Dr. Fazal Noorbasha (2012)." Design of 64-bit Peripheral Component Interconnect Bus at 66MHz".

[16] Altera Master Core IP Mt64 User Guide.

[17] On-Chip FIFO Memory Core in Volume 5: Embedded Peripherals of the Quartus II

[18] Memory System Design from Altera Corporation February 2010.

[19] Stratix Device Handbook, Volume 1 July 2005.

[20] Ali Azarian, Mahmood Ahmadi (2009). Reconfigurable Computing Architecture. 978-1-4244-4520-2/09 2009 IEEE.

AUTHORS PROFILE

G.Prasad (M-IEEE, FIETE) received M.Tech degree in Electronics from JNTU Hyderabad in 1995, MBA from IGNOU, New Delhi 2000. He is presently working as Scientist "SF" at National Remote Sensing Centre, ISRO, Hyderabad. His nature of work includes design and development of satellite data acquisition systems, high speed communication between different data acquisition sites through satellite networking. His research interests include VLSI designs, embedded system and realizing systems on programmable chips.

Copyright © 2014 CTTS.IN, All right reserved

N.Vasantha (M-IEEE, LM-CSI,IETE,ISTE, M VSI) received the B.E. degree from College of Engineering, Guindy, Madras, in 1977, the M.Tech. degree from JNTU, Hyderabad, AndhraPradesh, in 1986, the Ph.D. degree in Electronics & Communication Engineering from the Osmania University, Hyderabad, AP, in 2008.She is currently working as Professor & Head, Department of Information Technology, Vasavi College of Engineering, Hyderabad, AP. She is the recipient of the prestigious IETE-Prof. K.Sreenivasan's Memorial Award(2010).

Copyright © 2014 CTTS.IN, All right reserved