International Journal of Engineering and Science Invention (IJESI)
ISSN (Online): 2319-6734, ISSN (Print): 2319-6726
www.ijesi.org ||Volume 13 Issue 3 March 2024 || PP 06-11

Multiple designs and parallel processing in FPGA

Dr. G.Prasad
NRSC, ISRO, Hyderabad, India

ABSTRACT: *FPGAs are reprogrammable hardware chips for implementation of digital logic. Their reconfigurability is attributable to the presence of configurable logic slices and interconnects. The possibility of hardware configuration in FPGAs allows the designer to design multiple modules that run parallel and independently to each other. Few applications like interface logic to computer systems will require separate chips for the particular application thus increasing the chip count or requiring additional boards for the particular types of computer interfacing. These applications are traditionally met by a custom developed chips design, and it is usually a very expensive solution. This paper describes a new approach of reconfigurable logic in a single chip and meeting the interfacing requirements. This is possible due to the availability of the reconfigurable features in the FPGAs. Earlier data simulator design was specific for a satellite with no selection facility for the parameters. This paper discusses how FPGAs offer true hardware parallelism and deliberates on some areas which benefit from FPGAs' parallel processing such as onchip memory, interfacing to host systems for Data Acquisition and Processing and reconfiguring. To demonstrate the possibility and the consequent advantages of parallel processing, a data simulator and a data logger was designed in a Stratix FPGA.*

KEY WORDS - *Data Simulator, Data logger, FPGA, IP Core, PCI, Stratix.*

Date of Submission: 18-03-2024 Date of Acceptance: 31-03-2024

I. 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 [1]. 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. 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 pattern generators of different frequencies, parallel to serial converters, control logic and filters just to name a few of them. Reconfiguration allows placing a specific logics in different sections of the FPGA. Without dynamic reconfiguration you would either need a multiple FPGAs on a single board or multiple boards for different applications. The first case is a waste of FPGA area. The second case implies increased hardware costs and power consumption.. With dynamic reconfiguration an FPGA which has the capacity to house multiple design modules.

The sheer volume of information to be processed in today's world has risen exponentially with time. This has led to the development of faster processor cores with higher clock speeds. Now the paradigm is shifting

to multi-core processors. A good example of this would be Intel's gen4 and gen5 processors with multiple cores on a single chip.

Although parallel processing is possible using multiple cores, their cost and power consumption are major issues in their practical application. Achieving parallel processing using a single microprocessor with a single core is an avenue that has been extensively researched. Numerous pipelining algorithms have been developed to achieve the fastest throughput – however, the fact remains that a single processor can execute only one instruction at a time.

In contrast, FPGAs (Field Programmable Gate Arrays) can offer true parallelism in processing. They can be configured to have multiple processors functioning in parallel. FPGA processing is purely hardware based processing – there is no fetching of instructions required; the program description determines the configuration of the hardware itself i.e. the processing is implemented in hardware rather than software.

II. 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.

III. 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 [2]. 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. the hardware itself i.e. the processing is implemented in hardware rather than software.

IV. Design of Data simulator and Data Acquisition logic on the FPGA.

4.1 Data simulator

The Data Simulator is built in the FPGA to simulate the data pattern of selected satellite. Data Simulator logic generates the Frame Synchronization code, line count in the auxiliary field and video data which is a pattern. A Crystal Oscillator of 200MHz is the source which is used to generate the required frequency. The serial data and clock are connected to a RJ45 connector. Generic Data Format received to ground from payloads onboard will consist of the following elements.

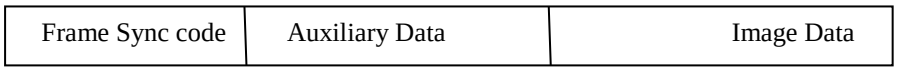

Figure 1. Generic Frame format of satellite data.

Depending upon the satellite configured the frame length is selected from the logic. The frame sync code which is a Pseudo random code is generated by a 7 bit shift register as shown in the code. The serial frame sync pattern is included in the first 128 bits of the frame. Auxiliary Data consists of the satellite health parameters which will be used for the ground level processing of the acquired image. There are several parameters in the aux data field. The line count is implemented using a counter, which counts the number of frames being transmitted from the Data Simulator. Apart from this many other parameters are simulated, serialized and included in the field identified for the Auxiliary Data. It is the reference for decommuting the data. The video or image data as acquired by the optics of the payload is stored in the Image data area. Hence a Generic Data Simulator is designed which will help in testing and simulating the previous and future satellites. The block diagram is shown in Fig below.

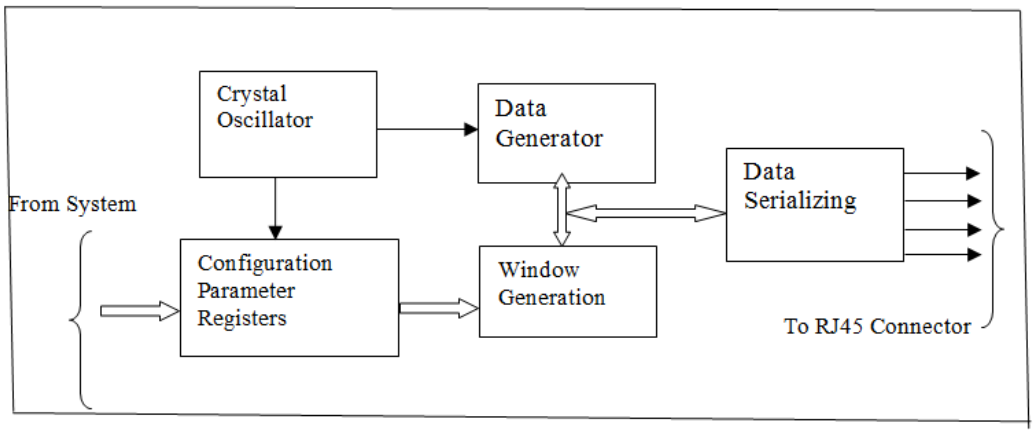

Figure 2. Block diagram of Data Simulator logic.

The data simulator design is partitioned as two Functions with different outputs. The first one is the channel clock generator which has its inputs from a crystal oscillator, passive reset circuit and user defined configuration parameters. This module gives as out the serial clock per channel depending on the number of channels. The second module is the simwnd module where the parameters for a satellite is selected and defined. The various parameters that can be selected are pixel clock width, length of the frame sync code, length of the auxiliary data field, length of the image field, total frame size, whether data should be randomized or not. The auxiliary data field consists of health parameters of the satellite and those also are selectable like channel identifier, line counter and this module will output serial data of the selected satellite and corresponding end of line pulse. A pseudo random code for frame sync code, dynamic parameters of the Auxiliary data and different patterns of the video data are generated. The data is suitably embedded into the frame as per the format and satellite selected, serialized and sent as output. Through the application program the parameters for a satellite selected are entered as inputs to the FPGA through the program, accordingly the parameters pertaining to the satellite selected are loaded. Data for two streams comprising of number of frame sync length, aux data length, video data length, total frame length, auxiliary data width, pixel width etc are generated. The output is the

simulated satellite data in serial form and the end of line indicator. This serial simulated data is the input for the data acquisition logic. The resources utilized for the data simulator logic are 545 logic elements and the logic is operating at 200MHz. Four I/O s were used to connect to the external devices. Earlier data simulator design were specific for a satellite with no selection facility for the parameters. In this design by selecting and loading the parameters the desired satellite data was simulated and used for testing. In the simulation waveform shown in Figure 3, the reference clock of 200MHz is the source and the parameters pertaining to a satellite is selected. The output waveforms are the Frame sync window, Auxilary window and Video window. The serialized data generated from the simulator is shown in fig which will be fed as input to the frame synchronization logic.

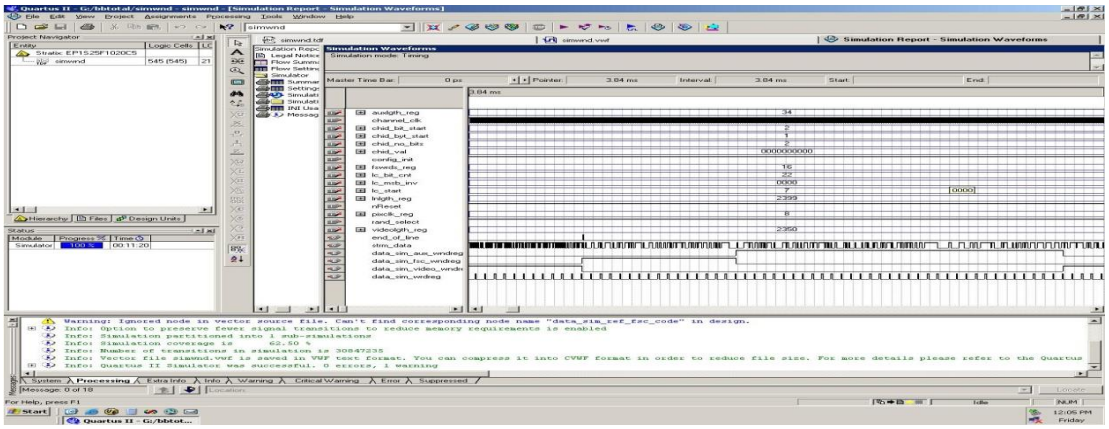

Figure 3. Outputs of Data simulator logic.

4.2 Data Acquisition logic

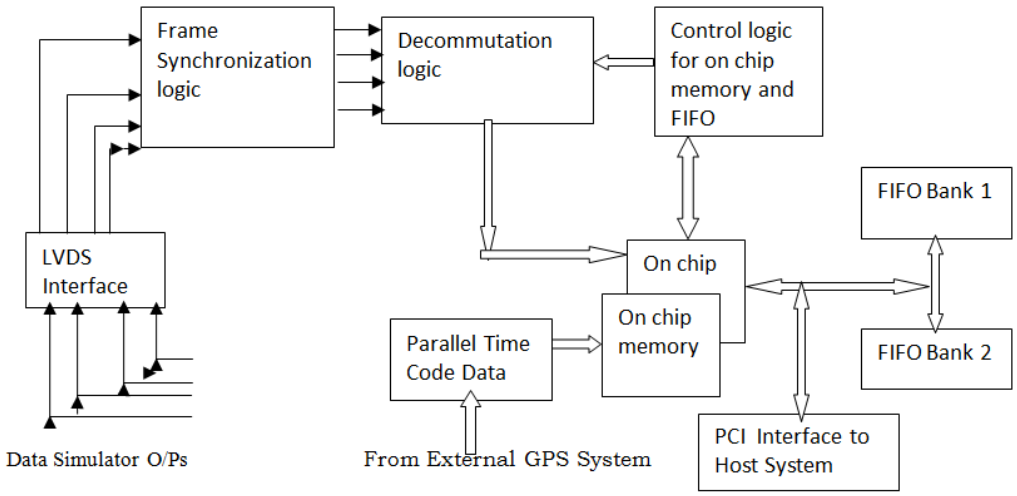

Figure 4 .Block Diagram of Data Acquisition Hardware

Data Acquisition systems are very crucial elements in satellite Ground stations. The principle objective is to develop generic satellite data acquisition hardware with all the related modules designed on a single FPGA and also ensuring the PCB form factor to be designed for housing into standard server class machines. ITANIUM processor is the target system to be used for testing. The modules designed for satellite data acquisition will Auto correlate the incoming satellite data, for detecting the valid frames received from the satellite and decommutate the data into 64bit Qwords and 64bit word clock. Writing the decommutated data into the on chip memory of the FPGA in suitable frames depending upon the satellite selected.GPS parallel time code data from a Time code unit is read out, synchronized and embedded into the data frame as an extra word. Integrated the on chip memory to an external FIFO for mass storage of the data on the Data Acquisition hardware. Interfaced the acquired data to Server through a PCI core for storage on the disk in real time and enabling further processing in near real time.

The Printer Circuit Board (PCB) housing the data acquisition and simulator logic modules is capable of handling PCM data of data rates up to 200Mbps per channel. It has advanced features to perform front end processing of remote sensing satellite data on a serial bit stream received from Bit Synchronizer and Signal

Conditioner (BSSC). The front end processing consists of dual frame synchronization for I and Q channels, generation of corresponding word clock from the serial clock, conversion of the serial data to parallel 64bit word form. Also Parallel Binay Coded Data (BCD) time code data from the GPS time code generator is tagged for every frame. The parallel data is first stored on the FPGA on chip memory and subsequently buffered into external FIFO bank made of dual 128k X 72 FIFOs [3,4]. The write and read control logic are generated within the FPGA. The parallel data is transferred to the ITANIUM II server through PCI Interface logic which is implemented within the FPGA [5,6]. A Data Simulator is also built in the FPGA for verifying and validating the logic during testing and maintenance. All the above mentioned modules are developed in a single latest state of art high density and high speed 90nm FPGA Stratix EP1S25F1020C5. The resources utilized for the data acquisition logic is 617 logic elements were utilized and the logic was operating at 200MHz. Seven I/O s were used to connect to the external devices. The same logic when compared with Correlator chips would require 4 standalone chips which have 96 I/O pins and occupies board space on 11mm x 11mm per chip and the frequency of operation is 50MHz maximum. 573456bits were utilized for implementing the on chip memory of 64bit width. The same logic when compared with Memory chips would require 16 standalone chips which have 120 I/O pins per chip.The write clock speed was varying between 2.5MHz to 4MHz and the read clock of the on chip memory was 15MHz.

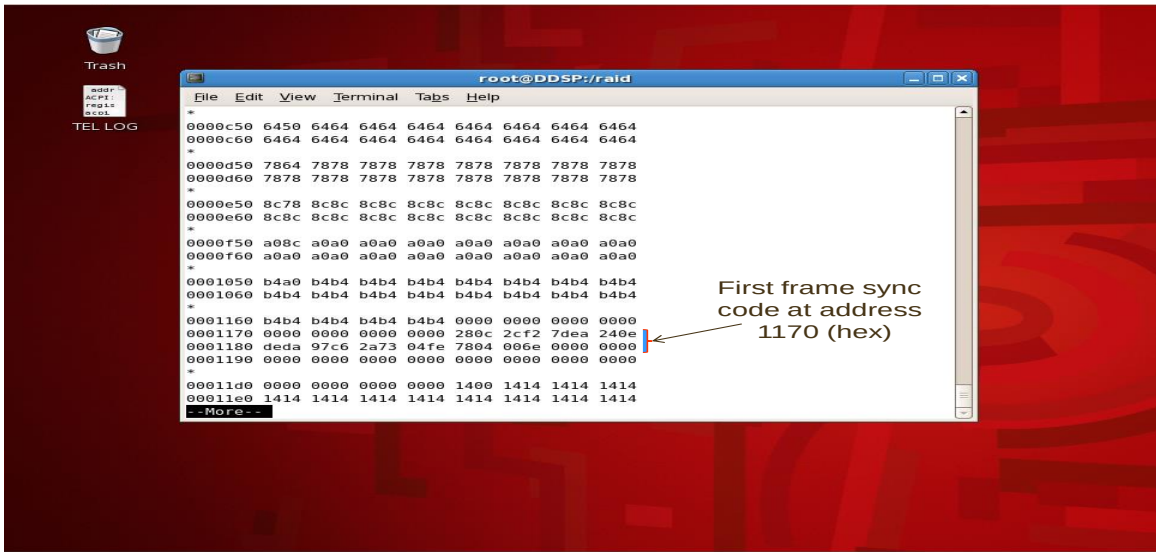

Figure 5. Data acquired from Data Acquisition Hardware Logic.

The Embedded Hardware is plugged into an Itanium Server and operates on Red hat linux [7]. The configuration is it uses Extended Parallel Instruction computing Architecture (EPIC). It has 8GB RAM, four processors and 4 PCI- X slots for housing the Embedded hardware. Two application programs are executed for the data simulator to generate data and application program for Data Acquisition is executed for data logging from the same card simultaneously. Data Acquired is shown in Figure 5. Hence multiple design and parallel processing is demonstrated.

V. Conclusion

A Generic Data Acquisition system has been developed to meet the past, present and future requirements of Satellite Data Acquisition. All the relevant modules have been developed on a single chip and PCB form factor is realized so that it will fit into any standard server class machines. Few applications like interface logic to computer systems will require separate chips for the particular application thus increasing the chip count or requiring additional boards for the particular types of computer interfacing. These applications are traditionally met by a custom developed chips design, and it is usually a very expensive solution. This paper describes a new approach of reconfigurable logic in a single chip and meeting the interfacing requirements. This is possible due to the availability of the reconfigurable features in the FPGAs. Earlier data simulator design were specific for a satellite with no selection facility for the parameters. Unlike the existing system where the time code data is embedded into the Frame sync code after 32bit and later the remaining bits of the frame sync code are compared, here a dedicated Qword for time code and status has been allocated, which helps in retrieving the raw data received from the satellite by just removing the extra word for play back and simulations. Interfaced the data from the FIFO to the cache memory of the Server through a PCI master core for storage on the disk for real time and near real time processing, and achieved a high throughput of 220 Mbytes/second.

All the modules are realized in the single FPGA, and the FPGA still has resources for modifications to cater for future satellites. Two such cards were housed in the server and tested so that in the future if number of input channels is increased, the same cards can be used with firmware changes. Two cards were installed in the server and tested data acquisition in both the cards simultaneously, a throughput of 210Mbytes/second was achieved. With firmware changes the same hardware can read data from the disc and serialize the data so as to work as data serializer for the validation and chain testing of the Satellite ground segment elements. Thus as high speed data acquisition system using an FPGA was successfully designed, developed and tested to prove that the modules required for high speed satellite data acquisition can be in an FPGA. Thus Data Acquisition system on a programmable chip has been achieved with the dual functions of Data Acquisition and Data Pattern Generator. Thus in future if data with 4 channel input have to be acquired it is possible with the same hardware and will call for changes in code. It will eliminate the requirement of second card for acquiring the additional 2 channels of data.

REFERENCES

[1]. Katherine Compton, Scott Hauck "An introduction to Reconfigurable computing", Northwestern University, Dept. of ECE Technical Report, 1999.

[2]. Stratix Device Handbook, Volume 1 July 2005. [20] Ali Azarian, Mahmood Ahmadi (2009). Reconfigurable Computing Architecture. 978-1-4244-4520-2/09 2009 IEEE.

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

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

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

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

[7]. 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.