PARSER: Bit Stream Analyzer and Data Extractor

September 18, 2000 Product Specification

CSELT S.p.A

Via G. Reiss Romoli, 274
I-10148 Torino, Italy
Phone: +39 011 228 5259
Fax: +39 011 228 5695
E-mail: viplibrary@cselt.it
URL: www.cselt.it

Features

- Independent of input data format (data with, cell length, spacing between packets) and from data throughput.
- Analysis of a wide range of data syntax, bit alignment, logic operations commonly used in most standards (IP, MPEG, ATM).
- Operations executed by the PARSER can be defined through a programming interface connected to a CPU/ Controller which can be a 4, 8, 16, or 32-bit processor.
- Multiple checks and data extraction can be performed on the same input data, using concurrent information processing, without affecting the speed of the operation.
- Extraction of a customizable number of data sets; each data set is checked independently and is sent to output with valid flag and bit mask. A mask error flag is also provided.
- PARSER operations do not modify the input bit stream. The data flow is transported to output without any change; data samples are synchronous with the analysis results.
- Core Customization:
  - Input (Output) Interface data width.
  - Programming Interface data width and bus retention.
  - Size and formatting of the program instruction word.
  - Mask definition method (mask compressed or expanded)
  - Program width (number of internal FIFO registers)
  - Maximum number of concurrent operations performed on a data sample.
  - Maximum number of data sets to be analyzed/ extracted.

AllianceCORE™ Facts

Core Specifics[1]		
Supported Family	Spartan	Virtex
Device Tested	S40-3	V50-6
CLBs[2]	477	548
Clock IOBs	2	2
IOBs[3]	86	86
Performance (MHz)	31	67
Xilinx Tools	M3.1i	M3.1i
Special Features	None	None
Provided with Core		
Document	User Manual	
Design File Formats	EDIF netlist, XNF netlist,	
Constraints Files	NCF constraints file	
Verification	VHDL testbench	
Instantiation Templates	VHDL, Verilog	
Reference Designs & Application Notes	None	
Additional Items	None	
Simulation Tool Used		
Synopsys VSS		
Support		
Design and customization support provided by CSELT		

Notes:
1. Data refer to the following customization:
   - 8 data in Input FIFO
   - 1 threshold Input FIFO
   - 4 input data bits
   - 15420 Forwarding Table Rows
   - 16 output data bits
   - 16 memory data bits
   - 18 memory address bits
   - 2 MEM_CYC bus bits
   - 0 level for MCE
   - 0 base memory address for the table

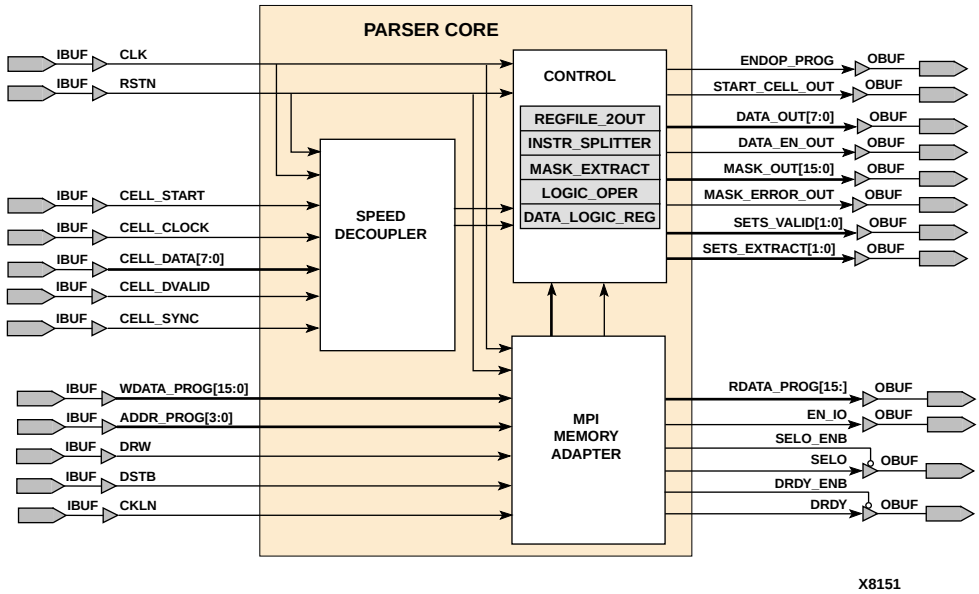

Figure 1: Parser Block Diagram

Applications

The parser has been defined to fit a variety of applications, such as ATM, Internet Protocol and MPEG2 transport stream.

General Description

PARSER is a general purpose intellectual property core used to analyze and extract data sets from a generic input bit stream using user data as reference.

The PARSER core analyzes and extracts sets of bit strings fields from a generic input bit stream. The incoming data flow must adhere to a standard protocol such that the core can identify the beginning of a new cell (or packet) and the presence of a valid data. The core has been defined to fit a variety of applications, such as ATM, Internet, and MPEG2 transport stream.

The PARSER recognizes any new cell or packet (by means of a CELL_START signal) and identifies the position of any data, updating a counter for any new data read from the input bit stream. The operations performed on each data are repeated for any new cell (or packet); these processes are driven by instruction set codes stored in the PARSER registers. The instruction set codes can be written by an external CPU through a generic microprocessor interface. The instruction set codes and the protocol of the microprocessor interface are described in the User Guide Manual.

The source code version of the core allows full customization of the core by means of VHDL generics. The parameters include the parallelism of the input bit stream, the maximum number of concurrent operations computed on the same data and the number of data sets to extract.

Functional Description

The internal architecture of the PARSER core is shown in Figure 1. A brief description of the operations of each module follows.

MPI Memory Adapter

The MPI Memory Adapter block acts as an adapter between the Microprocessor Interface and an internal synchronous memory block. The interface allows writing and reading the PARSER programming instructions and addressing data through a standard interface. Refer to the User Manual for instruction set codes.

Speed Decoupler

The Speed Decoupler block reads and samples the Input data flow, regenerating both data and the control signals of the input bit stream.

To avoid metastability the primary data inputs of the module are protected by a Dual Rank Synchronizer (DRS): data are double-sampled and synchronization is rebuilt on the PARSER clock.

Control

The Control block manages the data bit stream from the Speed Decoupler and the programming from the MPI Memory Adapter. It performs the data flow analysis and

field extractions according to the parameter values and the program instructions. The Control unit is composed of many sub-modules, as explained below.

REGFILE_2OUT

The Regfile_2out block stores the PARSER instruction set codes written by the CPU through the MPI Memory Adapter interface. It is implemented as a register file with two independent read ports, one accessible by the core side and one by the microprocessor interface).

Data Count

The Data Count block checks the current data sample and verifies whether it has to be processed, as specified in the PARSER instruction set. This Unit addresses the Regfile_2Out to read the instructions and to identify the presence of a field to be analyzed or extracted.

Instruction Splitter

The Instruction Splitter block collects information from the current instruction (addressed by the DATA_COUNT block) and distributes all the information to perform the parsing operations on the data sample, according to the parameters set in the PARSER.

Mask Extract

The Mask Extract block performs two operations: it expands the bit masks in the mask-compressed mode and cleans the masks from errors, generating an error flag.

Logic Operation

The Logic Operation block performs the operation check on the fields contained in the data sample, according to the bit masks and the program instruction. This Unit can be instantiated in multiple copies, to perform concurrent processing on many fields in the same data sample (this feature is controlled by means of parameters).

Data_Prog_Reg

The Data_Prog_Reg block introduces delays on many internal data and signals, to have a constant latency and a global synchronization between all the internal modules, independently from the PARSER features and behavior defined by parameter values.

PARSER Control

This unit is the kernel of the data processing. Analysis of the partial results and data extraction, with mask reordering, is accomplished collecting data from the Parser internal Units.

Pinout

The pinout of this core has not been fixed to a specific FPGA I/O allowing flexibility with a user's application. Signal names are shown in the block diagram in Figure 1 and described in Table 1.

Table 1: Core Signal Pinout

Signal	Signal Direction	Description
CLK	Input	Master clock
CLKN	Input	Asynchronous reset
RSTN	Input	Data input
CELL_DATA[7:0]	Input	Data input
CELL_CLOCK	Input	Data valid input
CELL_DVALID	Input	Forward or inverse transform selection input
CELL_START	Input	Start operation
CELL_SYNC	Input	Data output
DATA_OUT[7:0]	Output	Data synchronism output
START_CELL _OUT	Output	Data valid output
DATA_EN _OUT	Output	Forward or inverse transform selection output
START_CELL _OUT	Output	Output data read enable
DATA_EN_OUT	Output	Input field/frame FDCT mode
MASK_OUT[15:0]	Output	Output zig-zag/scan IDCT mode
MASK_ERROR _OUT	Output	Input zig-zag/scan IDCT mode
SETS_VALID[1:0]	Output	Clear output data request
SETS_EXTRACT [1:0]	Output	Input buffer memory status information
WDATA_PROG [15:0]	Input	Output buffer memory status information
RDATA_PROG [15:0]	Output	Processing status(active/ waiting for data)
ADDR_PROG[3:0]	Input	Programming interface output data bus
ENDOP_PROG	Input	Programming interface end operation
EN_IO	Output	Programming interface input/output data bus enable
SELO	Output	Programming interface valid data select
SELO_ENB	Output	SELO output enable
DRDY	Output	Programming interface data ready
DRDY_ENB	Output	DRDY output enable
DRW	Input	Programming interface data read/write
DSTB	Input	Programming interface data strobe

Core Modifications

The source code version of the PARSER core is parameterizable. Parameters are implemented as a set of generics in the synthesizable VHDL source code of the core. Parameters allow the user to specify some architectural and functional features of the synthesized core netlist, so as to adapt it to a specific design or application.

Table 2: Core Parameters (VHDL Generics)

Parameter	Description
PROG_WIDTH	Programming port width
DATA_WIDTH	Input data width
POS_FIELD_WIDTH	Position field width
MAX_PROG_WORDS	Maximum number of programming words
MAX_SPARE_FLAGS	Custom number of spare flags
MAX_SPARE_SETS	Custom number of spare sets
MAX_SETS	Maximum number of sets to extract data
MAX_OPER	Maximum number of concurrent operations
MASK_COMPRESS	=0: Mask expanded <>0: Mask compressed
BUS_RET	Bus retention: 1= Available

Verification Methods

Extensive functional simulation has been performed for different values of the core parameters, using the Synopsys VSS simulator. Simulation scenarios (including data and command files) and parametric test bench used for design verification are provided with the core.

The parametric test bench is composed of a parametric test vector generator and the PARSER cell.

Recommended Design Experience

Experience with the Xilinx design flow and logic system design is recommended to the users of the netlist version of the core. For the source code version, users should also be familiar with the Synopsys FPGA synthesis tools (VHDL Compiler, FPGA Compiler) and simulator (VSS).

Ordering Information

CSELT provides the PARSER core under license for use in Xilinx programmable logic devices. Please contact CSELT S.p.A. for information about pricing, terms and conditions of sale.

CSELT S.p.A. reserves the right to change any specification detailed in this document at any time without notice, and assumes no responsibility for any error in this document.

Related Information

Xilinx Programmable Logic

For information on Xilinx programmable logic or development system software, contact your local Xilinx sales office, or:

Xilinx, Inc.
2100 Logic Drive
San Jose, CA 95124
Phone: +1 408-559-7778
Fax: +1 408-559-7114
URL: www.xilinx.com

For general Xilinx literature, contact:

Phone: +1 800-231-3386 (inside the US)
+1 408-879-5017 (outside the US)
E-mail: literature@xilinx.com

Parser Implementation Request Form

To: CSELT S.p.A.

FAX: +39 011 228 5695

E-mail: viplibrary@cselt.it

From: _____

Company:_____

Address:_____

City, State, Zip:_____

Country:_____

Phone:_____

FAX:_____

E-mail:_____

CSELT configures and ships Xilinx netlist versions of the Arbiter core customized to your specification. Please fill out and fax this form so that CSELT can respond with an appropriate quotation that includes performance and density metrics for the target Xilinx FPGA.

Implementation Issues

1. Programming port width (4/8/16/32): _____

2. Input data bus width (4/8/16/32): _____

3. Maximum length of input sequence? _____

4. Number of programming words? _____

5. Number of fields to be extracted (1 to 8): _____

6. Maximum no. of concurrent operations (1 to 8): _____

Business Issues

1. Indicate time scales of requirement:
_____ date for decision
_____ date for placing order
_____ date of delivery

2. Indicate your area of responsibility:
_____ decision maker
_____ budget holder
_____ recommender

3. Has a budget been allocated for the purchase?
Yes _____ No _____

4. What volume do you expect to ship of the product that will use this core? _____

5. What major factors will influence your decision?
_____ cost
_____ customization
_____ testing
_____ implementation size

6. Are you considering any other solutions? _____