IPlogiCAM Internet Protocol Content Addressable Memory

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

- Core composed of a software control (C algorithm and software utilities) and a hardware block (VHDL source or netlist)
- C algorithm and software running on user's processor and C compiler
  - to generate the compact Forwarding Table starting
  - from an ASCII routing Table,
  - to insert and delete entries,
  - to write reports
  - to generate ASCII file for the IPlogiCAM testbench and the External Memory Table Dump
- Portable Standard C Algorithm
- Programmable Internet Protocol address length (including new IPv6 address length)
- Programmable number of rows in the forwarding table (stored in an external memory)
- Available memory protocols with different speeds (wait cycles)
- Programmable Input and Output Data Bus size
- Programmable Memory Data bus and Address bus width
- Programmable table location (starting address and length) in the external memory
- Automatic recovery after receiving a bad address or Table errors
- Fast and Flexible Search Algorithm with decision depth related to table size
- Parametric FIFO to store the input identifier; programmable buffer size and programmable threshold to signal when the stored data overcomes it
- Core requires a processor and a C compiler
- Core customization
  - Input data size

AllianceCORE™ Facts

Core Specifics[1]		
Supported Family	Spartan	Virtex
Device Tested	S10-3	V50-6
CLBs[2]	129	148
Clock IOBs	1	1
IOBs[3]	62	62
Performance (MHz)	31	49
Xilinx Tools	M3.1i	M3.1i
Special Features	None	None
Provided with Core		
Documentation		User Manual
Design File Formats		EDIF netlist, XNF netlist
Constraints File		NCF constraint file
Verification		VHDL testbench
Instantiation Templates		VHDL, Verilog
Reference Designs & Application Notes		None
Additional Items		C algorithm and software utilities
Simulation Tool Used		
Synopsys VSS		
Support		
Design and customization support provided by CSELT		

Notes:
1. Data refer to the following customisation:
   - 8-bit 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
2. Utilization numbers for Virtex are in CLB slices
3. Assuming all core I/Os are routed off-chip

Features (contd)

- Output Data size
- External RAM Data and Address size
- Customizable Memory Interface: external RAM Wait cycles, chip enable logic level
- Forwarding Table Mapping on the external memory (Physical Addresses)
- Input buffer (FIFO) size and threshold value to avoid FIFO congestion

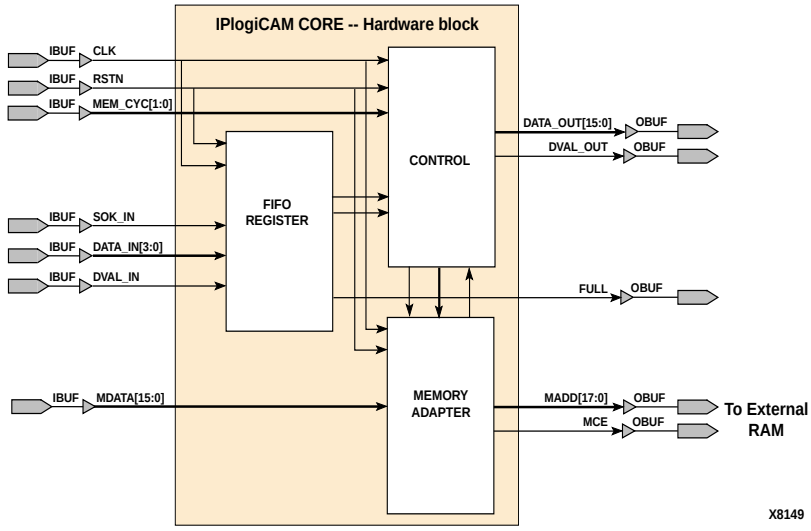

Figure 1: Hardware Block of the IPlogiCAM

Applications

This core is suitable for Internet Protocol routers where large network routing tables are involved.

General Description

The IPlogiCAM core implements an optimized algorithm for hardware assisted Internet Protocol address lookups.

The most time-consuming operation in forwarding the data streams towards the destination is the lookup of the destination address carried in each packet. Given a destination address, routing information must be retrieved in a large forwarding table consisting of a "dictionary" of initial address sub-strings (i.e. address prefixes). The problem is to find the longest initial sub-string of the destination address that is included in the forwarding database (longest prefix match). Internet Protocol longest prefix match is made slow and complex by the fact that the prefixes have variable length and can be prefixes of each other. The most obvious way to solve the longest match problem is to use a ternary CAM. The IPlogiCAM implements a logic ternary CAM realized by a "walking-tree" search algorithm operating on external SRAM. The search algorithm consists of C code that looks at the user's IP address table to create the search table which is then searched by the hardware block implemented in the FPGA.

Functional Description

The IPlogiCAM core is composed of a C-based control algorithm and a hardware block implemented in the FPGA. The control algorithm written in C code is used to prepare the search table that will be written on an external SRAM.

This C algorithm runs on the user's processor and a C compiler. The hardware block then performs the Longest Prefix Match operation efficiently, searching the data structure stored in the external SRAM, with minimal memory accesses (4 accesses and a 2-4 Mbyte memory are enough for today's largest existing table).

The user has to just set the length of input search key (e.g. an IP address) and he will obtain the next hop address. The user can also set the required trade-offs between speed (number of memory accesses) and memory size so as to achieve the optimal performance/cost ratio.

The internal architecture of the IPlogiCAM hardware block is shown in Figure 1. A brief description of the operation of each module follows.

FIFO Register

The FIFO Register Block receives and buffers the incoming Internet Protocol addresses. This block is composed of a set of registers that act as a FIFO; the registers can be independently read and written. The control block manages the reads and writes avoiding conflicts.

The input data are extracted from the Internet Protocol data flow and usually they arrive in the form of the burst with a long spacing between them. The FIFO buffer allows to decouple the incoming address rate with the internal core speed, mainly due to the external SRAM accesses.

The incoming data protocol is:

- SOK_IN goes active ('1') for any incoming new destination address (32 bit in IPv4, 128 bit in the IPv6)
- DATA_IN is a portion of the Internet Protocol address: You need 8 clock cycles to receive the 32-bit address.

The DATA_IN size is customizable and the chosen value has a big impact on the required external memory amount;

- DVAL_IN is the data valid signal and it is related to the address portion, in which the IP destination address is split. The input FIFO allows for non consecutive incoming portions

The FIFO register block stores the incoming addresses and sets the Full signal when the number of written (and not yet read) registers overcomes a threshold defined by the user.

The control block reads the FIFO, at a speed depending on the external SRAM cycle. When the FIFO is empty, the IPlogiCAM waits for new IP addresses.

Control

The Control block reads data from the FIFO and uses them to address the forwarding table stored in the external SRAM.

To meet the memory speed requirements without reducing the IPlogiCAM clock frequency, a programmable number of wait cycles, depending on the external memory access time, can be inserted. The number of Wait cycles can be set as low as zero to have single clock read access.

The Control block reads and translates the forwarding table stored in the SRAM. When a valid identifier for the Next Hop Address is found, according to the Longest Prefix match rule, it is sent to the core output (DATA_OUT) with a data valid signal (DVAL_OUT set to '1').

Memory Adapter

This block is used to generate the signals to control and address the external SRAM.

The forwarding table is stored on the external SRAM according to parameters that define the starting address. The size for Memory data bus and address bus can be customized, according to the dimension of the forwarding table stored in the SRAM.

The memory chip enable active level is also defined by a parameter, and the read cycle length is controlled by the value set in the MEM_CYC input bus (the maximum value allowed is defined with a generic).

The Memory Adapter block acts on the bus width and the address mapping, fitting the external memory characteristics with the parameter chosen values and with the forwarding table features.

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
RSTN	Input	Asynchronous reset
DATA_IN[3:0] (1)	Input	Parallel data input
DVAL_IN	Input	Parallel Data Valid Input
SOK_IN	Input	Start of Key
FULL	Output	Next Data Rejected
MADD[17:0] (2)	Output	External Memory Address
MDATA[15:0] (3)	Input	External Memory Data
MCE	Output	Memory Chip Enable
MEM_CYC[1:0] (4)	Input	Wait Cycle to Access Ext. Mem.
DATA_OUT[15:0] (5)	Output	Parallel Data Output
DVAL_OUT	Output	Parallel Data Valid Output

Notes:
1. Vector range depends on the DIM generic
2. Vector range depends on the MADD_SIZE generic
3. Vector range depends on the MDATA_SIZE generic
4. Vector range depends on the MAX_WAIT generic
5. Vector range depends on the OD_SIZE generic.

Core Modifications

The source code version of the IPlogiCAM core is parametric. 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
NUM_DATA	Number of Data Storable in Input FIFO
THR_FULL	Threshold for FULL signal from Input FIFO
DIM	Input Data Width
MAX_ROW	Max Number of Forwarding Table Rows
OD_SIZE	Output Data Size
MDATA_SIZE	Memory Data Size
MADD_SIZE	Memory Address Size
MAX_WAIT	MEM_CYC Bus Width
MCE_VAL	Valid Level for MCE
BASE_ADD	Base Memory Address for the table

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 IPlogiCAM 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 S.p.A. provides the IPlogiCAM 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

IPlogiCAM
Implementation Request Form

To: CSELT S.p.A.

FAX: +39 011 228 5695

E-mail: viplibrary@cselt.it

CSELT configures and ships Xilinx netlist versions of the IPlogiCAM 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.

From: _____

Company:_____

Address:_____

City, State, Zip:_____

Country:_____

Phone:_____

FAX:_____

E-mail:_____

Implementation Issues

1. Input Data size (4, 8, 16)?: _____

2. External RAM Data size? _____

3. External RAM Address size? _____

4. External RAM Wait Cycles (0, 1, 2)? _____

5. External RAM Chip enable logic level?

Number of Forwarding Table entries?

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? _____