XILINX®

XAPP204 (v1.2) May 2, 2000

Using Block RAM for High Performance Read/Write CAMs

Author: Jean-Louis Brelet

Summary

CAM (Content Addressable Memory) offers increased data search speed. In various applications based on CAM, there are differing requirements for data organinzatation and read/write performance. The innovative design described in this application note is suited for small embedded CAMs with high-speed match and write requirements. The reference design is built using the true Dual-Port™ block SelectRAM+ feature of Virtex™ FPGAs. Application note XAPP201, "An Overview of Multiple CAM Designs in Virtex Devices", discusses the diverse solutions available when implementing CAM while introducing the specific solution described in this application note.

Introduction

The block SelectRAM+™ memory built into the Virtex devices can be used as a 16-word deep by 8-bit wide (16 x 8) CAM using the innovative design techniques described in this application note. A reference design (see Appendix A) provides parameterizable Verilog and VHDL code to cascade several block RAMs configured as 16 x 8 CAM. CAM speed is equivalent to the access time of a Virtex block RAM for a single clock cycle match (read), and a one or two clock cycles write.

Medium size CAMs can be implemented in Virtex slices with different design techniques. Three approaches to designing CAM in Virtex devices and CAM size and performance comparisons are discussed in XAPP201.

CAM16x8 Macro

Features of the CAM16x8 include:

- 128 bits
- 16-word x 8-bit organization
- Independent match (read) and write data input busses
- Decoded address output (or 16-bit "one-hot" decoded address)
- Fully synchronous match port (or read port)
- Fully synchronous write port
- Single clock cycle match (single or multi-matches)
- Single clock cycle write (and single clock cycle erase)
- Match Enable input
- Write Enable input
- Write or Erase mode (assert to “one” if write, to “zero” if erase)
- Reset match port (forces the output bus to zero synchronously)
- Cascadable
- Initialization during device configuration (empty by default)
- Fits in one Virtex block SelectRAM+ memory

Figure 1 shows a CAM16x8 macro built on the True Dual-Port block SelectRAM+ memory. Write port A is independent of read port B. Both ports are fully synchronous relative to their respective clock.

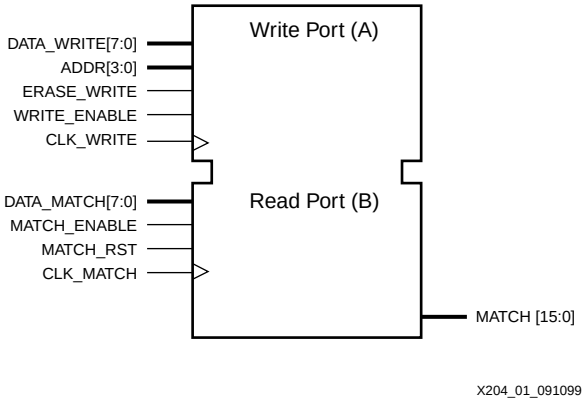

Figure 1: **CAM 16 x 8 Macro**

Virtex FPGAs facilitate different CAM sizes based on the number of block RAMs in each device. Using the cascadable CAM16x8 macro, Table 1 shows possible combinations of CAM organization in Virtex devices.

Table 1: **Maximum CAM width and depth in Virtex Devices**

Virtex Device	Number of Block SelectRAM	Maximum CAM size	Maximum CAM width	Maximum CAM depth
XCV50	8	1,024 bits	16 x 64-bit	128 x 8-bit
XCV100	10	1,280 bits	16 x 80-bit	160 x 8-bit
XCV150	12	1,536 bits	16 x 96-bit	192 x 8-bit
XCV200	14	1,792 bits	16 x 112-bit	224 x 8-bit
XCV300	16	2,048 bits	16 x 128-bit	256 x 8-bit
XCV50E	16	2,048 bits	16 x 128-bit	256 x 8-bit
XCV400	20	2,560 bits	16 x 160-bit	320 x 8-bit
XCV100E	20	2,560 bits	16 x 160-bit	320 x 8-bit
XCV600	24	3,072 bits	16 x 192-bit	384 x 8-bit
XCV800	28	3,584 bits	16 x 224-bit	448 x 8-bit
XCV200E	28	3,584 bits	16 x 224-bit	448 x 8-bit
XCV1000	32	4,096 bits	16 x 256-bit	512 x 8-bit
XCV300E	32	4,096 bits	16 x 256-bit	512 x 8-bit
XCV400E	40	5,120 bits	32 x 160-bit	640 x 8-bit
XCV600E	72	9,216 bits	32 x 288-bit	1,152 x 8-bit
XCV1000E	96	12,288 bits	64 x 192-bit	1,536 x 8-bit
XCV405E (EM)	140	17,920 bits	64 x 280-bit	2,240 x 8-bit
XCV1600E	144	18,432 bits	64 x 288-bit	2,304 x 8-bit
XCV2000E	160	20,480 bits	128 x 160-bit	2,560 x 8-bit
XCV2600E	184	23,552 bits	128 x 184-bit	2,944 x 8-bit
XCV3200E	208	26,624 bits	128 x 208-bit	3,328 x 8-bit
XCV812E (EM)	280	35,840 bits	128 x 280-bit	4480 x 8-bit

Because CAM output is a decoded address (similar to a 16-bit "one-hot" decoded address), the depth is expandable without additional logic. Each location has a single address bit output.

When data is present at a particular address, the corresponding address line goes High. When data is not present, the address line is Low. A decoded address output allows multiple matches, *i.e.* , the same data might be found in several locations in only one clock cycle. The write mode is similar to writing 8-bits of data in one of the 16-word x 8-bit memory blocks.

Figure 2 shows the read mode of a 64-word x 8-bit CAM built on four block SelectRAM+ primitives. The 64-bit decoded address bus is generated by merging four 16-bit decoded addresses.

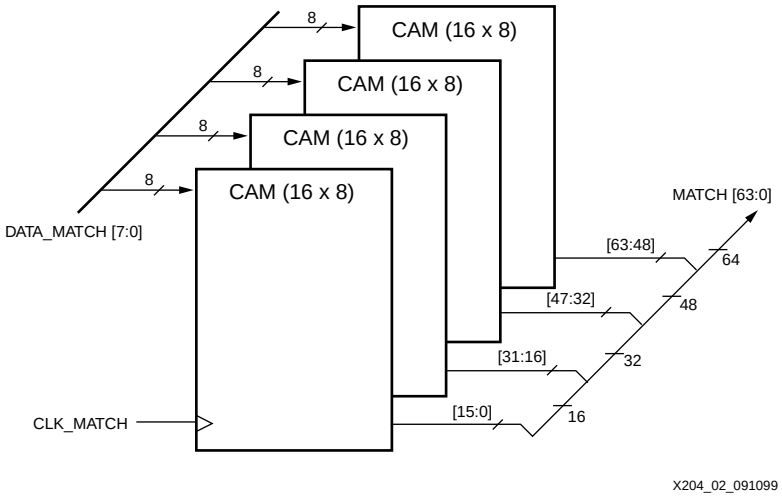

Figure 2: **CAM 64-word x 8-bit in Read Mode**

In a write mode, a 6-bit bus is used as an address for the 64 x 8 CAM. This address bus is composed of a 4-bit bus to each set of CAM16x8 address inputs and a 2-bit bus decoded to select one of the CAM blocks.

To write 8-bits of data to one of the 64 locations (6-bit address bus), one of the four CAM16x8 blocks is selected with the two MSBs of the 6-bit address bus. The four LSBs are connected to the regular address inputs ADDR[3:0] of each block.

The width of the design is also expandable with extra AND gates. Each CAM16x8 contains only 8-bits of the total word width. Two 16 x 8 CAMs allow for a 16-bit width, with 8 bits stored in the first CAM16x8 block and 8 bits in the second block. A match is found only if both 8-bit locations match the specified 16-bit data. A 2-input AND gate for each CAM output signal provides the final decoded address. Figure 3 shows a16-word x 16-bit CAM in read mode, built with two block SelectRAM+ primitives

Because the basic CAM16x8 macro has a "one-hot" decoded address, both CAM depth and width are simultaneously expandable to build the desired CAM size while providing the same input/output features.

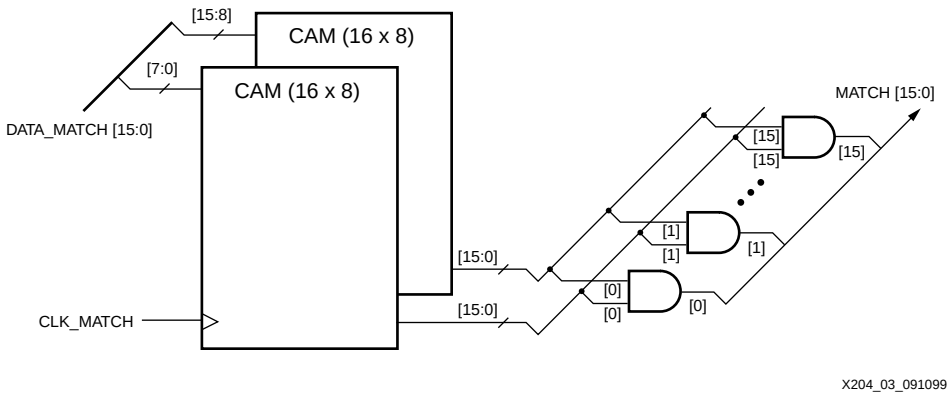

Figure 3: **CAM 16-word x 16-bit in Read Mode**

To achieve an optimal performance when cascading CAM16x8 blocks, the number of block SelectRAM+ memories per Virtex column has to be considered. Each Virtex device has two block SelectRAM+ columns. The number of blocks per column depends on the Virtex device.

The Table 2 shows some CAM sizes using only one Virtex column resulting in high-speed embedded CAM blocks.

Table 2: **CAM Sizes in Select Virtex Devices**

Virtex Device	Number of Block RAMs per Column	Number of Columns	CAM Size per Column	CAM Blocks per Column
XCV50	4	2	512 bits	64 x 8-bit
XCV50E	4	4	512 bits	64 x 8-bit
XCV300	8	2	1,024 bits	128 x 8-bit
XCV300	8	2	1,024 bits	64 x 16-bit
XCV300E	8	4	1,024 bits	64 x 16-bit
XCV405E	10	14	1,792 bits	64 x 28-bit
XCV812E	14	20	2,560 bits	80 x 32-bit
XCV1000	16	2	2,048 bits	128 x 16-bit
XCV1000	16	2	2,048 bits	64 x 32-bit
XCV1000E	16	6	2,048 bits	64 x 32-bit
XCV1600E	18	8	2,304 bits	72 x 32-bit
XCV2000E	20	8	2,560 bits	80 x 32-bit

Applications

An embedded CAM block is useful in many applications including networking designs and data processing. A small CAM can provide instantaneous protocol detection, parallel processing choices, or networking and router filtering.

A multi-protocol application typically requires different treatments of an input data stream. A selection is made based on the protocol type, generally encoded in a data header. An embedded CAM detects, in one clock cycle, the right protocol by searching an array of possible protocols. The designer must store all these protocols in the CAM block and can dynamically modify this list. When a protocol is found in the CAM, the corresponding address selects further operations.

Parallel processing or data recognition can benefit from embedded CAM blocks as well. Because the search for a matching data is performed in one clock cycle, even if multi-matches are found, CAM blocks accelerate those applications.

When the application uses a large data width, selected data bits (a smaller data width) are often sufficient to search the CAM block. An associated block RAM can store and retrieve the complete data. The scaleable size of the CAM blocks in Virtex devices and the flexible features offer many solutions to the designer. The concurrent access to search for data in a single clock cycle guarantees a high-speed result.

General Description

The unique Virtex block RAM approach is used to build the CAM16x8 block. This methodology is based upon the True Dual-Port feature of the block SelectRAM+ memories. Ports A and B can be configured independently, anywhere from 4096-word x 1-bit to 256-word x 16-bit. Each port has separate clock inputs and control signals. The internal address mapping of the block SelectRAM+ memory is the primary feature in designing a CAM in a True Dual-Port block RAM. Each port accesses the same set of 4096 memory locations using an addressing scheme dependent on the port width. Application Note XAPP130, "Using the Virtex Block SelectRAM+ Features", details the physical RAM location addressed for a particular width. Table 3 shows the low-order address mapping for a 1-bit and a 16-bit port width.

Table 3: **Block SelectRAM+ Port Address Mapping**

Port Width	Port Addresses															
1	4095...	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
16	255...	00															

This design technique configures port A as 4096-word by 1-bit wide and port B as 256-word by 16-bits wide. The block SelectRAM+ primitive in this particular configuration is named RAMB4_S1_S16. Each port contains independent control signals. Port A is the CAM write port, and port B is the CAM read or match port. Both the read and write CAM ports are fully synchronous and have dedicated clock and control signals.

Decoded 8-bit data in a Block RAM

An 8-bit data has 2^8 (256) possible values. A classic RAM stores the 8-bit data into an 8-bit location. However the 8-bit data can also be represented as a 256-bit word, with all zeros and a single “one” at the nth location, where n corresponds to the position given by the decoded 8-bit data. For example, if the data is "0000....0111" (a decimal seven), the decoded 256-bit word is “0000....000010000000”, where the “one” is at the seventh location counting from zero. Sixteen 256-bit words store sixteen decoded 8-bit words. As shown in Figure 4, an array of 16 x 256 represents 16 addresses from 0 to 15.

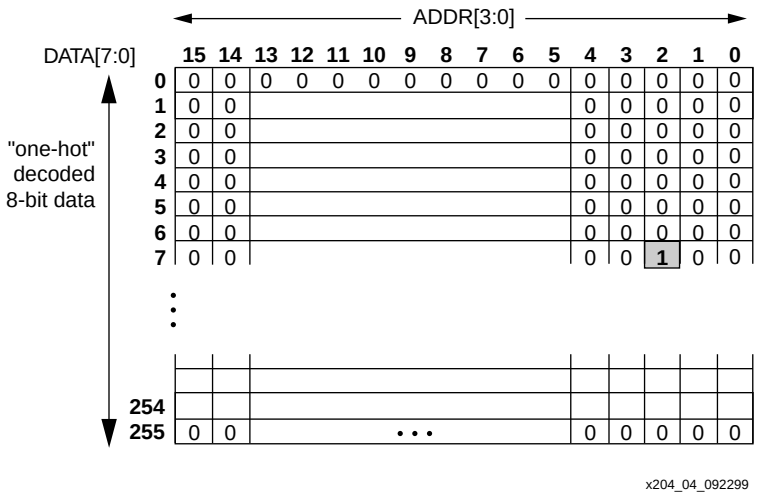

Figure 4: **CAM16x8 Port Address Mapping**

Finding a Match

The 4K-bit RAMB4_S1_S16 with a 16-bit wide port (port B) generates 16 values simultaneously. If the 8-bit data (DATA_MATCH) to be searched is connected to the 8-bit address (ADDRB) of port B as shown in Figure 5, the 16-bit port B generates the matches concurrently. Using the fact that a particular location corresponds to the decoded 8-bit data, the matching operation is equivalent to searching 16 locations for specific 8-bit data at the same time. Figure 5 shows the RAMB4_S1_S16 port B configured as a CAM read port.

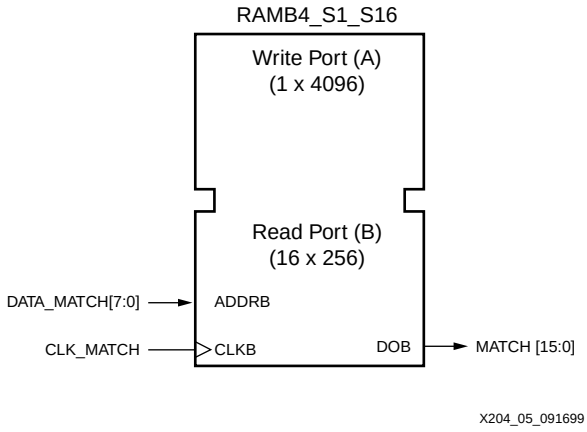

Figure 5: **Read Port of a CAM16x8 in a SelectRAM+ Block**

Reading into the CAM16x8 Example

If the data "0000 0111" was previously written at the address "0010" of the CAM16x8, a match operation is equivalent to a read operation of a block RAM with the 8-bit data "0000 0111" placed on the address input of port B (ADDRB[7:0]). The 16-bit port B output is "0000 0000 0000 0100" corresponding to a match found at location 2, as shown in Figure 6. Port B output is the decoded CAM address bus. If no match is found, the output is "0000 0000 0000 0000". If one or several matches are found, each corresponding location equals “one”. In this last case, the same 8-bit data has been stored at different addresses, and the CAM16x8 output is multi-matches.

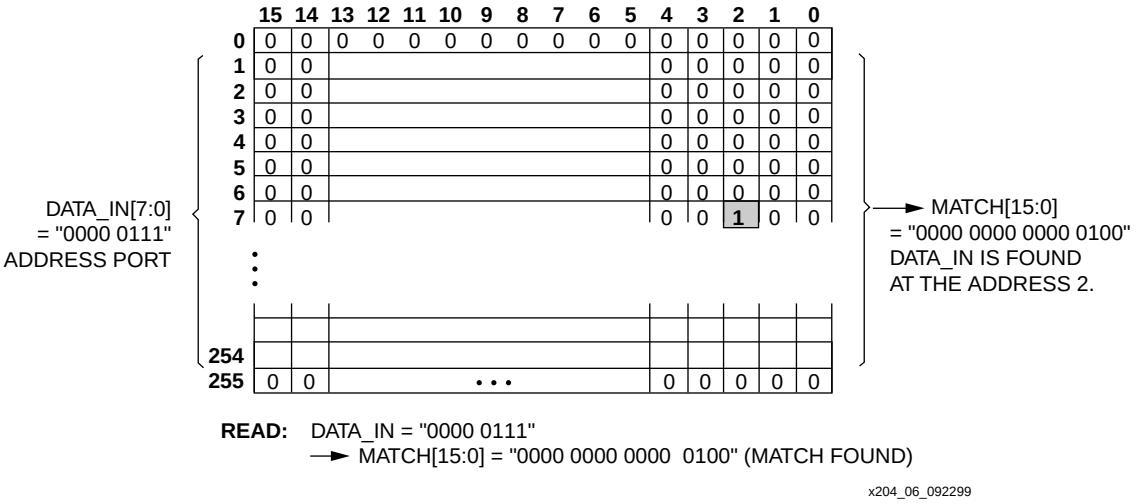

Figure 6: **An Example of a Read Into the CAM16x8 Block**

Writing into the CAM16x8

The CAM write port inputs are an 8-bit data bus, an address bus (four bits to address the 16 locations), control signals, and the clock. The 4-bit address bus selects a memory location. Writing new data into this location is equivalent to decoding the 8-bit data into a 256-bit "one-hot" word and storing the 256-bit word. However, if the CAM16x8 macro is initialized to zero, only one bit of the 256-bit word has to toggle. The location of the “one” is determined by the "one-hot" decoded 8-bit value.

Taking into account that the address port of the block SelectRAM+ primitive decodes the address bus, both operations previously described are combined into a simple write in the block RAM.

Port A configured as 4096 x 1 has a 1-bit data input and a 12-bit address input. The data input is asserted to “one”, and the 8-bit data plus the 4-bit address are merged in a single 12-bit address input.

With the 8-bit data as MSB and 4-bit address as LSB, the resulting 12-bit address input decodes the 8-bit data and selects one of the 16 memory locations simultaneously. The clock edge stores a “one” at the corresponding location. The Figure 7 shows the write port of the 6 x 8 CAM in a block SelectRAM+ primitive.

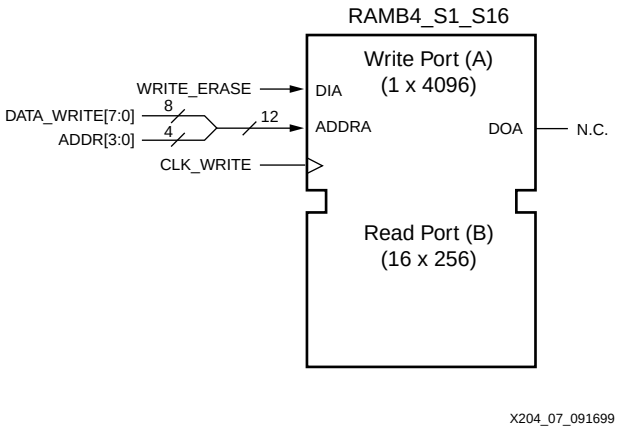

Figure 7: **Write Port of the CAM16x8 in a SelectRAM+ Block**

Erasing Data

To erase previously stored data, the selected location must be initialized to “zero” (256-bit word). The basic option is to write “zero” during 256 clock cycles by incrementing the 8-bit MSB of the address input from 0 to 255 . The 4-bit LSB of the address input is fixed and is used to select the CAM location.

The solution presented in the reference design requires a single clock cycle. The erase operation is equivalent to the write operation with the exception that a “zero” is stored instead of a “one”. The port A address input is again a combination of the 8-bit data to be erased and the 4-bit address bus. To perform the correct selection, the 8-bit data must be stored into a separate RAM block. The reference design shows a methodology using eight SelectRAM+ blocks (16 x 1 RAM in eight LUTs) to memorize the 16 words.

The choice between a no erase mode (the application doesn't need to delete the CAM data) and an erase mode is open to the designer. In the second case, two solutions offer different trade-offs between speed vs. Virtex slice area).

Performance of the 16 x 8 CAM16x8

A single clock cycle generates the 16-bit decoded address. A match operation on the CAM16x8 block is only a read operation into the SelectRAM+ block, and has the same performance. The write operation uses the standard SelectRAM+ write mode. The Virtex series data sheets detail the SelectRAM+ timing characteristics. Figure 8 shows the basic implementation of the CAM16x8 macro, built on a RAMB4_S1_S16 primitive.

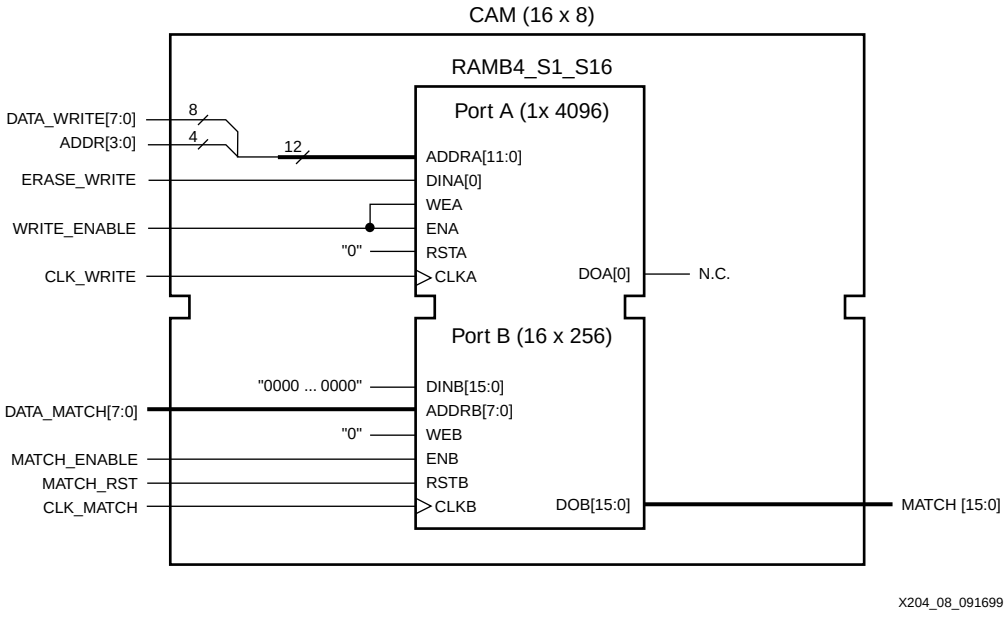

Figure 8: **CAM16x8 Macro Built on a RAMB4_S1_S16 Primitive**

Design Overview

The reference design described in this application note has an adjustable word depth. The output of the basic CAM16x8 macro is a decoded address generated in a single clock cycle. The number of output lines equals the number of words in the CAM. It is a multiple of 16, which represents the number of words in each CAM16x8 macro.

This reference design proposes some modules for generating the encoded output address. Because of the hierarchical VHDL code structure, it is easy to use this module for various CAM sizes according to the designers need along with a choice between LUT and TBUF implementations. A wide OR gate of all the decoded addresses creates a match flag asserted to one when a match is found.

Write Operation

This reference design implements a one clock cycle erase plus a one clock cycle write solution. The data stored in the CAM is also stored in RAM16x1 primitives and read back to erase the CAM. Figure 9 shows the write/erase configuration.

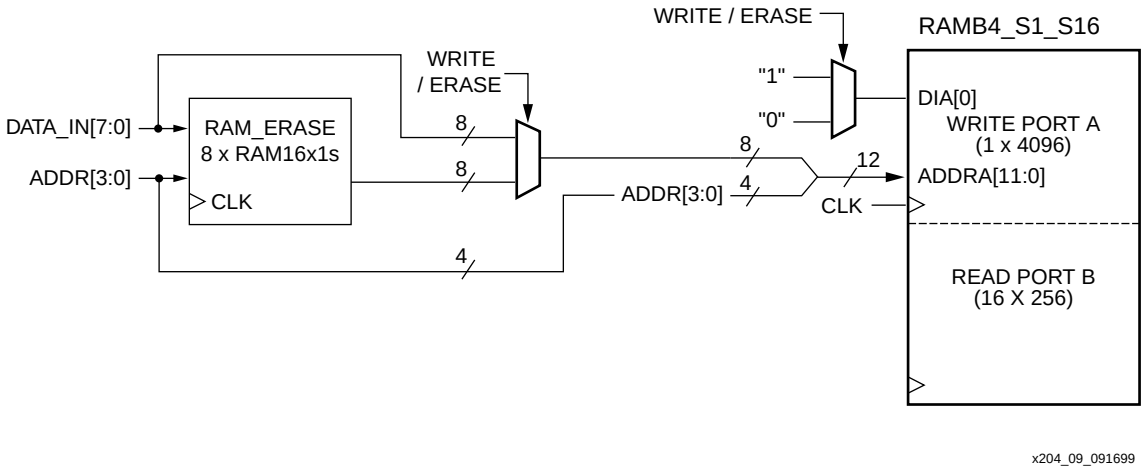

Figure 9: **CAM16x8 One Clock Cycle Erase and Clock Cycle Write**

Read Operation

The read or match operation is performed on port B. The decoded addresses are encoded in either a full logic implementation in a LUT, or a 3-state buffer TBUF implementation.

Built on the outputs of the CAM16x8 macro, the reference design describes how to generate a match signal and how to encode the CAM address output. Figure 10 shows a CAM16x8 with an additional wide OR gate generating a match flag. The MATCH_SIGNAL is asserted if one of more matches are found.

The 16-bit decoded address bus (MATCH) is encoded in a 4-bit address bus (MATCH_ADDR), which represents a valid address only if a single match is found. The reference design proposes to generate an additional signal, asserted to “one” when a single match occurs.

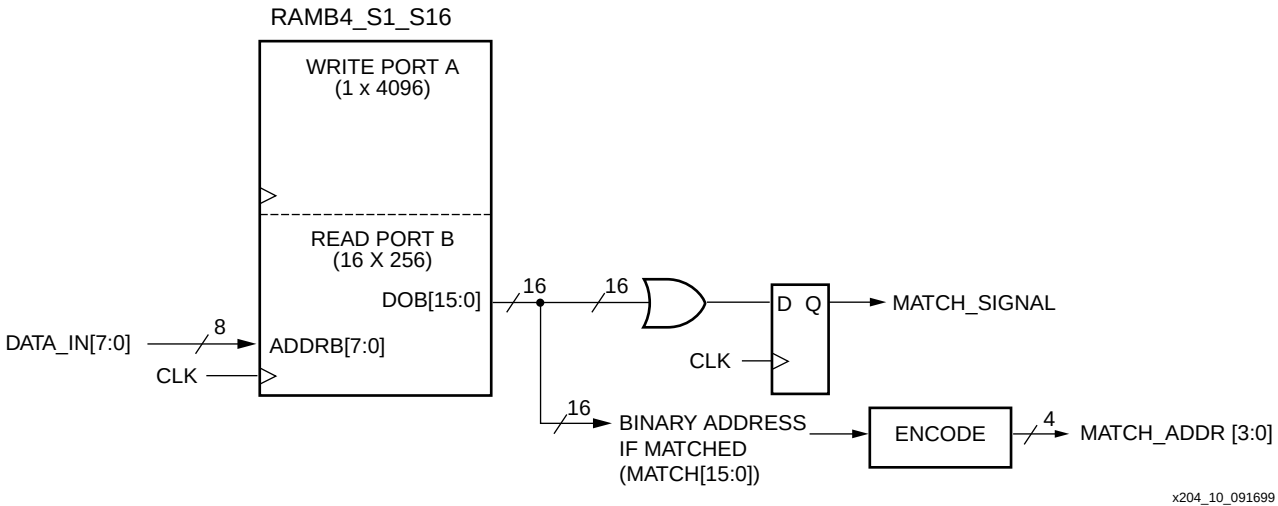

Figure 10: **CAM16x8 with a Match Flag and Encoded Outputs**

Designing a CAM16x8 in Virtex Block SelectRAM+ Primitives

With this application note and the reference design file XAPP204.zip, a designer can implement a fast read and write CAM module. The VHDL or Verilog code offers many possibilities to adapt this example to specific applications.

Features

- High performance single clock cycle match (read)
- High performance single clock cycle write or erase
- Generic "nb_cam16x8s" defining the number of CAM16x8 macros. The CAM depth is a multiple of 16. The smallest required Virtex device depends on the number of block SelectRAM+ memories (see the Table 1).
- Generic "addr_width" defining the number of address lines directly tied to the "nb_cam16x8s". A couple of examples are a 32-word CAM requiring five address lines and a 64-word CAM requiring six address lines.

Basic Building Block: CAM

The basic building block CAM16x8 is built on the block SelectRAM+ primitive as explained in the general description. In this design, only one 8-bit data bus is used on both the write port and the read port (DATA_IN[7:0]), and both ports share the same clock. The VHDL or Verilog code instantiating the block SelectRAM+ primitive is named Init_RAMB4_S1_S16 and provides an initialization example. By default, the CAM is empty, initialized to zeros.

Associated with this first module, a second basic building block is named Init_8_RAM16x1s. This module instantiates the eight RAM16x1 primitives to store the 16 words in a standard RAM and is used in the reference design to erase a location of the CAM16x8 in one clock cycle. As with the Init_RAMB4_S1_S16 module, the VHDL or Verilog code provides initialization (all zeros by default). The designer should initialize both modules with the same data for correct functionality.

The CAM_RAMB4 module instantiates the two previous modules to create a complete building block. The CAM_RAMB4 can find a match in one clock cycle, and can erase and write data in two clock cycles. This VHDL or Verilog module could be used in any HDL design as a building block.

Figure 11 shows the CAM_RAMB4 module built on one SelectRAM+ block and eight RAM16x1s. The multiplexer between the DATA_IN to be written and the DATA_WRITE to be erased is implicit in this implementation. (See Figure 9). Because the first clock cycle is the erase mode, the old data is read from the Init_8_RAM16x1s output DATA_WRITE. It becomes the 8-bit MSB address of ADDRA input. The new data (DATA_IN) is written into the Init_8_RAM16x1s and is reflected on the output DATA_WRITE. The second clock cycle is the write cycle with the new data automatically used as the eight MSB address, see Figure 12. The ADDR[3:0] input is unchanged during the two clock cycles and is used as the four LSB address.

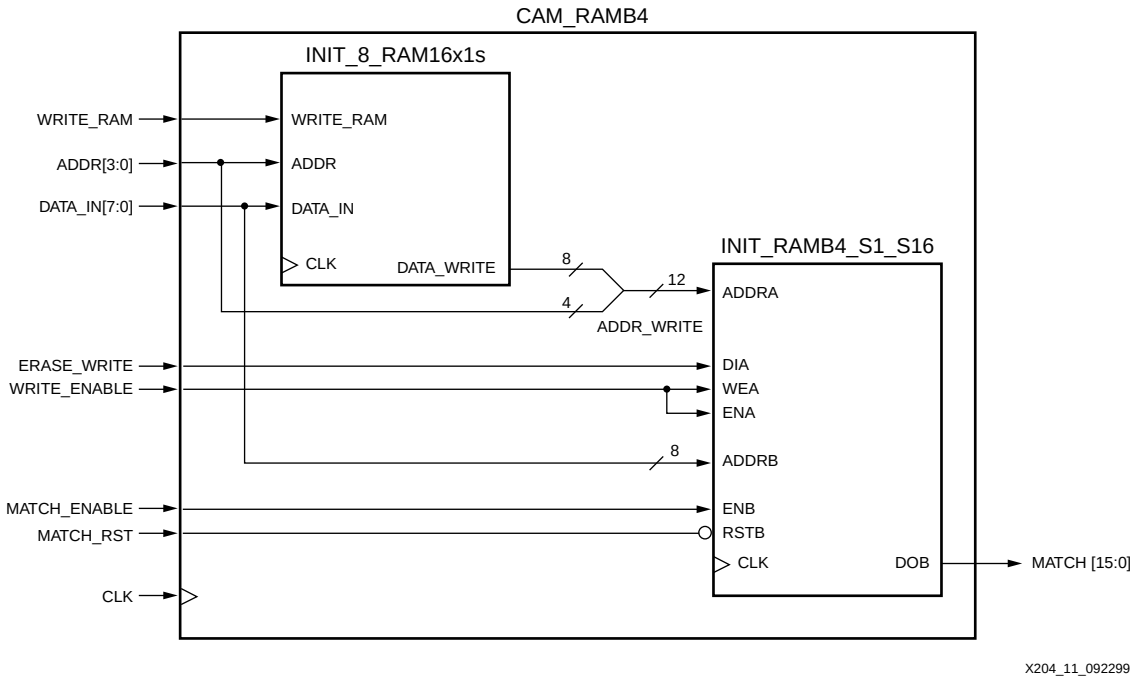

Figure 11: **CAM 16-word by 8-bit in one Block SelectRAM+ memory and eight RAM16x1s**

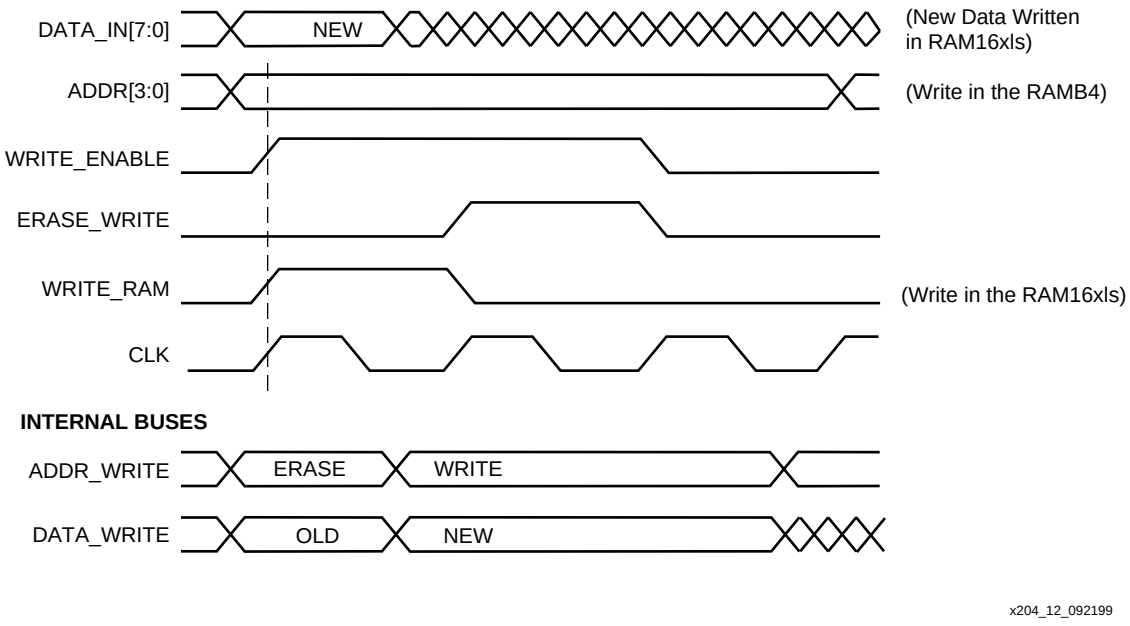

Figure 12: **Erase and Write Waveforms**

Module: CAM_generic_8s

To build the desired CAM size, several CAM_RAMB4 are instantiated in the CAM_generic_8s module, Figure 13. The CAM depth is a generic value "nb_cam16x8s", defining the number of CAM16x8s to be used. The CAM depth is a multiple of 16. A single CAM_RAMB4 requires a 4-bit encoded address output. The generic value "addr_width" corresponds to CAM depth. A 32-word CAM requires five address lines, a 64-word CAM requires six address lines, etc. The MATCH busses (CAM_RAMB4 outputs) are encoded to generate both the match address output MATCH_ADDR and a MATCH_OK signal (High when a match is found). This module is

similar to the XAPP203 reference design, (CAM_generic_word module). In the XAPP203 reference design, the basic building block is based on Virtex slices instead of the SelectRAM+ blocks. The basic module CAM depth is 16 words in both cases, sharing the encoder modules and the top level designs.

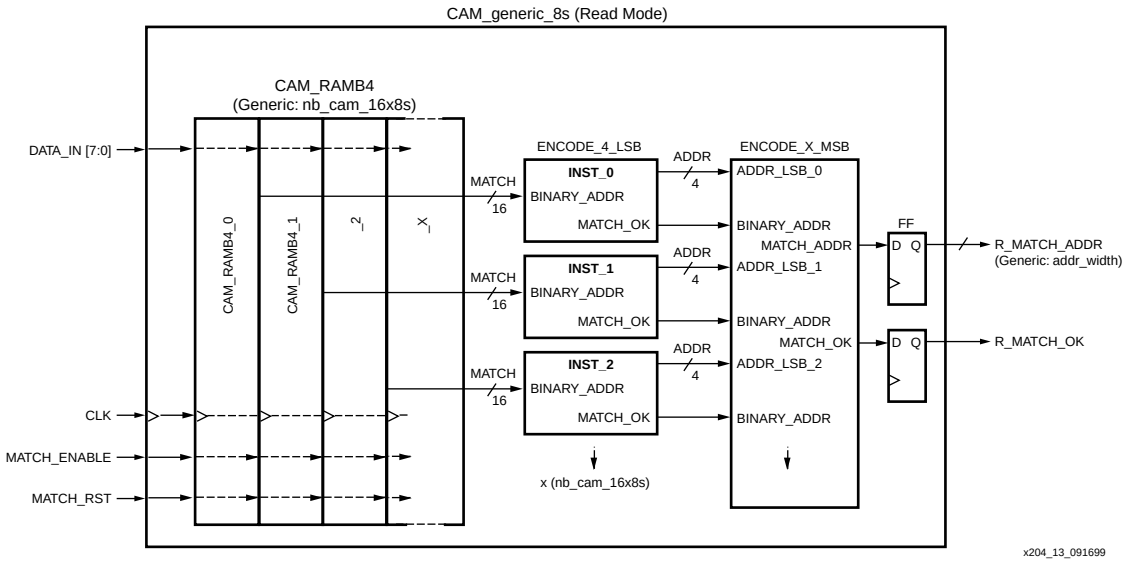

Figure 13: **CAM_generic_8s Read Path**

Module: CAM_top

This module is the top-level wrapper of the generic CAM (Figure 14). It registers all input and output signals.

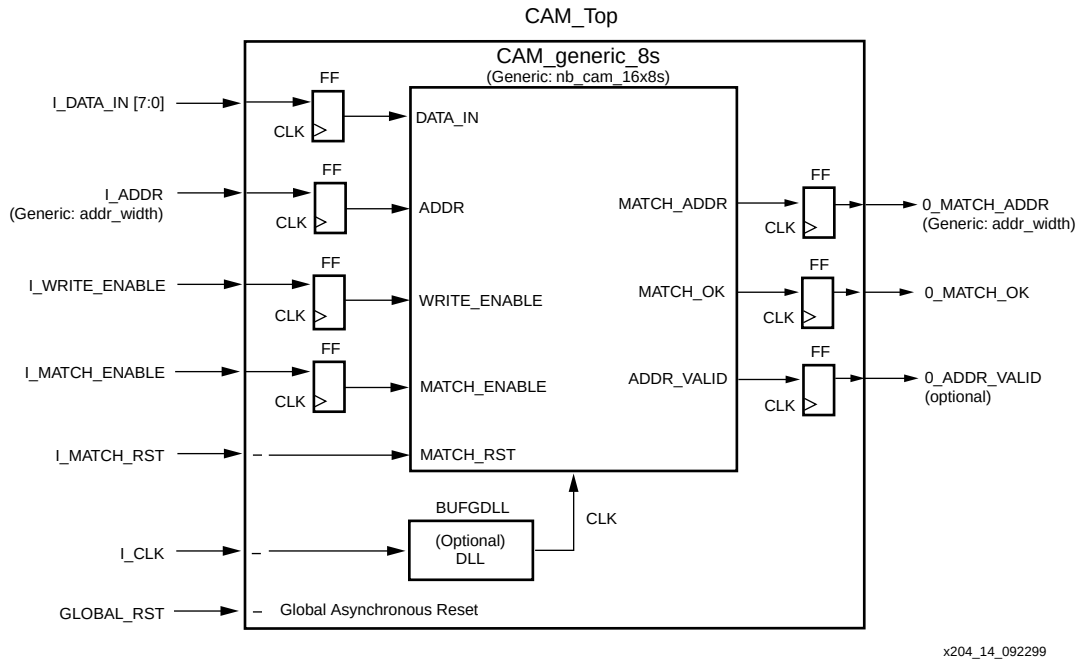

Figure 14: **CAM Design Top Level Block Diagram**

Pinout: (all I_XXX signals are inputs, O_XXX are outputs)

- I_DATA_IN is the 8-bit data input bus used by both read and write operations.
- I_ADDR is the generic input address bus used only to write new data into a selected location.
- I_WRITE_ENABLE is a two clock cycle active High signal (Erase then Write)
- I_MATCH_ENABLE enables a read access (active High)
- I_CLK is the clock. As an option, it can be routed through a DLL.
- O_MATCH_ADDR is the generic output address valid only in read mode.
- O_MATCH_OK is High when a match is found (read operation).
- O_ADDR_VALID is an optional signal when multiple matches occur. Active high if singles match occurred.

A recommendation when synthesizing this reference design is to keep the hierarchy for eventual floorplanning and to facilitate static timing analysis. A constraint file (UCF) should define the clock period. For optimal performance, the UCF files can constraint the location of each block SelectRAM+ memory in a column and the eight RAM16x1s primitives in the adjacent Slices column.

This reference design can be easily adapted to different CAM widths. As an example, four CAM16x8 macros produce a16-word by 32-bit CAM. The 16 additional 4-input AND gates used to generate the global decoded address fit in 16 LUTs.

Conclusion

The unique Virtex block SelectRAM+ True Dual-Port features allow an innovative technique to implement embedded fast CAMs. The CAM16x8 macro provides single clock cycle "one-hot" decoded address, even if several macros are cascaded to generate deeper or wider CAM blocks. The small size (16-word by 8-bit) of the basic CAM16x8 macro offers flexible choices in CAM size.

The reference design demonstrates a complete solution including an encoded address output, a match flag, and in addition to fast read access, a fast 2- clock-cycle erase/write access. Expandable CAM depth provides real flexibility.

The CAMs accelerate memory data search. Because of the various requirements among CAM-based applications, solutions must be flexible. In addition to the block SelectRAM+ solution, other approaches based on Virtex slices are presented in the application notes XAPP202 "Content Addressable Memory (CAM) in ATM Applications" and XAPP203 "Designing Flexible, Fast CAMs with Virtex FPGAs".

Appendix A: Synthesizable VHDL and Verilog Code Reference Design

This appendix describes a hierarchical, synthesizable design implementing an 8-bit word width and a parametric memory depth, CAM in Virtex devices. The complete HDL and Verilog code is available as a reference design (File: xapp204.zip or xapp204.tar.z).

Module: CAM_top.vhd

```
--
--     Module:   CAM_Top / Top Level
--     Design:   CAM_Top
--     VHDL code:    Hierarchical wrapper
--            Instantiate CAM_generic_8s (depth variable by 16x8bits word)
--
--     Synthesis Synopsys FPGA Express ver. 3.2 - Option = Preserve Hierarchy
--     Use of "pragma synthesis_off/on" and attributes
--
--     Description: Instantiate a CAM implementation
--     Registered inputs and outputs (CAM internal timing analysis)
--
--     Device:   VIRTEX Families
--
--     Created by: Jean-Louis BRELET / XILINX - VIRTEX Applications
--     Date: July 29, 1999
--     Version: 1.0
--
--     History:
--            1.
--
-- Disclaimer:THESE DESIGNS ARE PROVIDED "AS IS" WITH NO WARRANTY
-- WHATSOEVER AND XILINX SPECIFICALLY DISCLAIMS ANY
-- IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR
-- A PARTICULAR PURPOSE, OR AGAINST INFRINGEMENT.
--
--  Copyright (c) 1999 Xilinx, Inc.  All rights reserved.
----------------------------------------------------------------------
```

Module: CAM_generic_word.vhd

```
--
-- Module:    CAM_generic_8s
-- Design:    CAM_Top
-- VHDL code: Hierarchical RTL
--     Instantiate CAM_RAMB4
--     Instantiate INIT_RAMB4_S1_S16
--     Instantiate INIT_8_RAM16x1s
--     Instantiate ENCODE_4_LSB
--     Instantiate DECODE_X
--     Instantiate ENCODE_X_MSB
```

```
--     Choice the right ENCODE and DECODE modules according to the
       number of CAM16x8s
--     If "nb_cam16x8s" = 2 then ENCODE_1_MSB and DECODE_1 must be
       used: 1 bit to decode 2 CAM16x8s
--     If "nb_cam16x8s" = 4 then ENCODE_2_MSB and DECODE_2 must be
       used: 2 bits to decode 4 CAM16x8s
--     If "nb_cam16x8s" = 8 then ENCODE_3_MSB and DECODE_3 must be
       used: 3 bits to decode 8 CAM16x8s
--     If "nb_cam16x8s" = 16 then ENCODE_4_MSB and DECODE_4 must be
       used: 4 bits to decode 16 CAM16x8s
--     Note: Configuration is not supported by synthesis tools
--
-- Synthesis Synopsys FPGA Express ver. 3.2 - Option = Preserve Hierarchy
--     Use of "pragma synthesis_off/on" and attributes
--
-- Description: Instantiate "nb_cam16x8s" CAM_RAMB4 (see generic)
--     "nb_cam16x8s" x 16 words depth x 8 bits width
--     1 clock cycle Read (or Match),
--     2 clock cycles Write (Erase on the first clock then Store on the second)
--     If only 1 clock cycle => Erase Only.
--     MATCH_OK indicates one or more matches is/are found.
--     MATCH_ADDR ouput the address of the match, if ONLY ONE is found
--     ADDR_VALID indicates when MATCH_ADDR is a valid address (Optional)
--
-- Device: VIRTEX Families
--     modules CAM_RAMB4 fits in 1 BlockRAM column (+ CLB)
--     If "nb_cam16x8s" = 4 then CAM64x8s (fits in 1 XCV50 BlockRam Column)
--     If "nb_cam16x8s" = 8 then CAM128x8s (fits in 1 XCV300 BlockRam Column)
--     If "nb_cam16x8s" = 16 then CAM256x8s (fits in 1 XCV1000 BlockRam Column)
...
----------------------------------------------------------------------
```

Module: CAM_RAMB4.vhd

```
--
-- Module:    CAM_RAMB4
-- Design:    CAM_Top
-- VHDL code:    Hierarchical RTL
--               Instantiate INIT_RAMB4_S1_S16
--               Instantiate INIT_8_RAM16x1s
--
-- Synthesis     Synopsys FPGA Express ver. 3.2
--               Use of "pragma synthesis_off/on" and attributes
--
-- Description: Basic building block of a CAM using Select BlockRAM
--           16 words depth x 8 bits width
--           1 clock Read (or Match),
              2 clock Write (Erase on the first clock then Store on the second)
```

```
--           If NO match is found the output MATCH<15:0> = "0000000000000000"
--           MATCH bus gives on 16 signals a binary address.
--           Initialized RAM16x1s and RAMB4 in low level module
--
-- Device:   VIRTEX Families
...
----------------------------------------------------------------
```

Module: INIT_RAMB4_S1_S16.vhd

```
--
-- Module:   INIT_RAMB4_S1_S16
-- Design:   CAM_Top
-- VHDL code:VIRTEX primitives instantiation
--
-- Synthesis  Synopsys FPGA Express ver. 3.2
--           Use of "pragma synthesis_off/on" and attributes
--
-- Description:  Basic building block of a CAM using Select BlockRAM
--               16 words depth x 8 bits width
--               Instantiation RAMB4_S1_S16
--               Initialization of RAMB4: attributes to constraint PAR and
                 simulation
--
-- Device:   VIRTEX Families
--           1 RAMB4
...
----------------------------------------------------------------
```

Module: INIT_8_RAM16x1s.vhd

```
--
-- Module:   INIT_8_RAM16x1s
-- Design:   CAM_Top
-- VHDL code:VIRTEX primitives instantiation
--
-- Synthesis  Synopsys FPGA Express ver. 3.2
--           Use of "pragma synthesis_off/on" and attributes
--
-- Description:Basic building block of a CAM using Select BlockRAM & SelectRAM
--     Instantiate 8 RAM16x1s_1 for ERASE operation
--     Initialization of RAM16x1s: attributes to constraint PAR and simulation
--
-- Device:   VIRTEX Families
--           8 x SelectRAM16x1s_1
...
----------------------------------------------------------------
```

Module: ENCODE_4_LSB.vhd

```
--
-- Module:    ENCODE_4_LSB
-- Design:    CAM_Top
-- VHDL code: RTL / Combinatorial
--
-- Synthesis  Synopsys FPGA Express ver. 3.2
--            Use of "pragma synthesis_off/on" and attributes
--
-- Description:  Encode a 16 bits binary address into 4 bits and find if a
      match occurs
--     if BINARY_ADDR = "0000000000100000" => MATCH_ADDR = "0101"/ MATCH_OK = 1
--     Optional ADDR_VALID = 1 when only one Match (If simultaneous matches can
      occur)
--     However, the ADDR_VALID generation double the size of the combinatorial
      logic !
--     if no match found => MATCH_OK = 0 / ADDR_VALID = 0
      (MATCH_ADDR is not a valid address)
--     if 2 or more matches found => MATCH_OK = 1 / ADDR_VALID = 0
      (MATCH_ADDR is not valid address)
--
-- Device:    Virtex Families
-- ...
----------------------------------------------------------------
```

Module: ENCODE_4_MSB.vhd

This module is available in one bit ENCODE_1_MSB (CAM 32 words), two bits ENCODE_2_MSB (CAM 64 words), three bits ENCODE_3_MSB (CAM 128 words) and four bits ENCODE_4_MSB (CAM 256 words). Additional modules for other CAM depths could be easily created following this model.

```
--
-- Module:     ENCODE_4_MSB
-- Design:     CAM_Top
-- VHDL code:  RTL / Combinatorial
--
-- Synthesis   Synopsys FPGA Express ver. 3.2
--             Use of "pragma synthesis_off/on" and attributes
--
-- Description: Encode a 16 bits binary address into 4 bits, map with the LSB
      address and find if a match occurs
--     if BINARY_ADDR = "0000000000100000" => MATCH_ADDR = "0101"/ MATCH_OK = 1
--     Optional ADDR_VALID = 1 when only one Match
      (If simultaneous matches can occur)
--     However, the ADDR_VALID generation double the size of the
      combinatorial logic !
--     if no match found => MATCH_OK = 0 / ADDR_VALID = 0
      (MATCH_ADDR is not a valid address)
--     if 2 or more matches found => MATCH_OK = 1 / ADDR_VALID = 0
      (MATCH_ADDR is not valid address)
--
--     Choice between GATES ONLY implementation or BUFT
      implementation. (See comments)
--     Note: synthesis tools do not support Configuration
--
-- Device:     Virtex Families
-- ...
----------------------------------------------------------------
```

Module: DECODE_4.vhd

This module is available in one bit DECODE_1 (CAM 32 words), two bits DECODE_2 (CAM 64 words), three bits DECODE_3 (CAM 128 words) and four bits DECODE_4 (CAM 256 words). Additional modules for other CAM depths could be easily created following this model.

```
--
-- Module:    DECODE_4
-- Design:    CAM_Top
-- VHDL code: RTL / Combinatorial
--
-- Synthesis  Synopsys FPGA Express ver. 3.2
--            Use of "pragma synthesis_off/on" and attributes
--
-- Description: Decode 4 bits address into 16 binary bits
--            Generate an ENABLE bus
--
-- Device:    Virtex Families
-- ...
----------------------------------------------------------------
```

Similar modules ENCODE_3_MSB, ENCODE_2_MSB, ENCODE_1_MSB, DECODE_4, DECODE_3, DECODE_2, and DECODE_1 are available in the reference design files.

```
End of Appendix A.
```

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
9/23/99	1.0	Initial Xilinx release.
10/1/99	1.1	Minor corrections to text.
5/2/00	1.2	Updated for Virtex-E Extended Memory