Продолжение, начало в № 2`2007

Разработка базовых компонентов цифровых устройств, реализуемых на базе ПЛИС FPGA фирмы Xilinx®, с помощью генератора параметризированных модулей CORE Generator

Валерий ЗОТОВ
walerry@km.ru

Генерация элементов контекстно-адресуемых (ассоциативных) запоминающих устройств на основе параметризированного модуля Content Addressable Memory версии v5.1 с помощью средств CORE Generator

Запоминающие устройства с контекстной адресацией часто используются в составе кэш-памяти различных вычислительных систем. Для подготовки описаний элементов контекстно-адресуемых (ассоциативных) запоминающих устройств в составе генератора параметризированных модулей CORE Generator предусмотрено ядро Content Addressable Memory. Данный параметризированный модуль входит в состав группы ядер, предназначенных для генерации элементов запоминающих устройств *Memories & Storage Elements*. Последней версией этого ядра к моменту подготовки настоящей статьи являлась модификация Content Addressable Memory v5.1, которая позволяет формировать описания элементов ассоциативных запоминающих устройств, предназначенных для реализации на основе ПЛИС следующих семейств: Spartan-2, Spartan-2E, Spartan-3; Virtex; QPRO Virtex Rad-Hard; QPRO Virtex Hi-Rel; Virtex-E; QPRO Virtex-E Military; Virtex-II; Virtex-II PRO и Virtex-4.

Основные особенности параметризированного модуля Content Addressable Memory версии v5.1:

- возможность формирования описаний элементов оперативной и постоянной (Read Only CAM) контекстно-адресуемой памяти;
- широкий диапазон выбора разрядности входного слова данных в создаваемых описаниях элементов ассоциативных запоминающих устройств (от 1 до 512 разрядов);
- поддержка достаточно большого размера адресного пространства памяти, позволяющего формировать элементы контекстно-адресуемых запоминающих устройств емкостью от 16 до 4096 слов (фактическая емкость создаваемого элемента ассоциативной памяти ограничена только объемом физических ресурсов кристалла ПЛИС, выбранного для его реализации);
- возможность генерации описаний элементов ассоциативных запоминающих устройств, реализуемых на основе ресурсов блочной памяти ПЛИС Block SelectRAM или сдвиговых регистров SRL16 по выбору разработчика;
- возможность инициализации содержимого ячеек создаваемых элементов ассоциативной памяти данными из указанного файла;
- поддержка режима одновременной записи и чтения данных в создаваемых элементах контекстно-адресуемых запоминающих устройств (по выбору разработчика);
- возможность выбора одного из трех вариантов представления адресов ячеек, содержащих искомые данные, на выходной шине формируемого элемента ассоциативной памяти;
- применение (по выбору пользователя) выходных регистров, позволяющих добиться повышения производительности создаваемых элементов контекстно-адресуемых запоминающих устройств, реализуемых на базе блочной памяти ПЛИС Block SelectRAM;
- возможность выбора одного из двух режимов маскирования считываемых данных в элементах ассоциативной памяти, реализуемых на базе сдвиговых регистров SRL16;
- возможность применения дополнительного входа разрешения выполнения операций чтения и записи данных в формируемых элементах контекстно-адресуемых запоминающих устройств по выбору пользователя.

Выбор организации и способа реализации ассоциативного запоминающего устройства, формируемого на основе рассматриваемого параметризированного модуля, осуществляется с помощью соответствующего «мастера» настройки параметров, который содержит две диалоговые панели. В стартовой диалоговой панели «мастера», вид которой представлен на рис. 128, указывается название создаваемого элемента памяти, тип ресурсов ПЛИС, используемых для его реализации, разрядность входных шин данных, емкость, способ организации выходной шины адреса, а также некоторые дополнительные параметры конфигурации.

Для определения типа ресурсов кристалла, которые используются для реализации генерируемого элемента контекстно-адресуемого запоминающего устройства, необходимо воспользоваться двумя кнопками с зависимой фиксацией, расположенными во встроенной панели *Memory Type*. Чтобы сформировать описание элемента ассоциативной памяти,

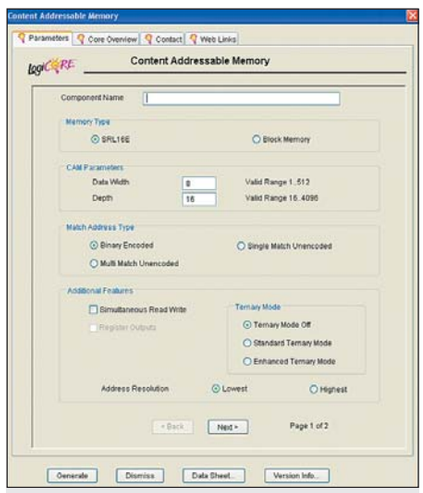

Рис. 128. Вид стартовой диалоговой панели «мастера» настройки параметров генератора элементов ассоциативной памяти Content Addressable Memory версии v5.1

реализуемого на базе сдвиговых регистров SRL16, следует зафиксировать в нажатом состоянии кнопку SRL16E. Если необходимо подготовить описание элемента контекстно-адресуемого запоминающего устройства, предназначенного для реализации на основе блочной памяти ПЛИС Block SelectRAM, то нужно нажать кнопку *Block Memory*.

Чтобы указать информационную емкость формируемого элемента ассоциативной памяти и разрядность порта (портов) записи и чтения данных, нужно воспользоваться полями редактирования, которые находятся во встроенной панели *CAM Parameters* (рис. 128). Разрядность порта (портов) записи и чтения данных задается в поле редактирования *Data Width*. Количество ячеек памяти в генерируемом контекстно-адресуемом запоминающем устройстве определяется с помощью поля редактирования *Depth*. На основании значения этого параметра автоматически вычисляется количество разрядов входной шины адреса записи в формируемом элементе ассоциативной памяти. Разрядность выходной шины адреса ячеек, содержащих информацию, соответствующую комбинации кодов входных данных и маски, определяется автоматически, исходя из значения, указанного в поле редактирования *Depth*, и выбранного формата представления адреса совпадения. Указываемые значения параметров *Data Width* и *Depth* не должны выходить за границы предельно-допустимых диапазонов, которые приведены в соответствующих строках *Valid Range*.

Выбор способа представления адреса ячеек, содержимое которых соответствует комбинации кодов входных данных и маски, на выходной шине создаваемого элемента ассоциативной памяти осуществляется с помощью группы кнопок с зависимой фиксацией, которая расположена во встроенной панели *Match Address Type* (рис. 128). Когда в нажатом состоянии зафиксирована кнопка *Binary Encoded*, адрес ячейки, содержимое которой совпадает с совокупностью значений, представленных на входных шинах данных и маски, будет отображаться на выходной шине формируемого элемента в виде двоичного кода, разрядность которого определяется следующим выражением:

$$K = \log_2(Depth), \qquad (33)$$

где K — разрядность выходной шины адреса чтения данных; *Depth* — количество ячеек памяти в создаваемом элементе контекстно-адресуемого запоминающего устройства.

Если содержимое нескольких ячеек соответствует комбинации значений, представленных на входных шинах данных и маски, то на выходной шине будет воспроизводиться код самого младшего или самого старшего адреса совпадения, в зависимости от выбранного значения параметра *Address Resolution*.

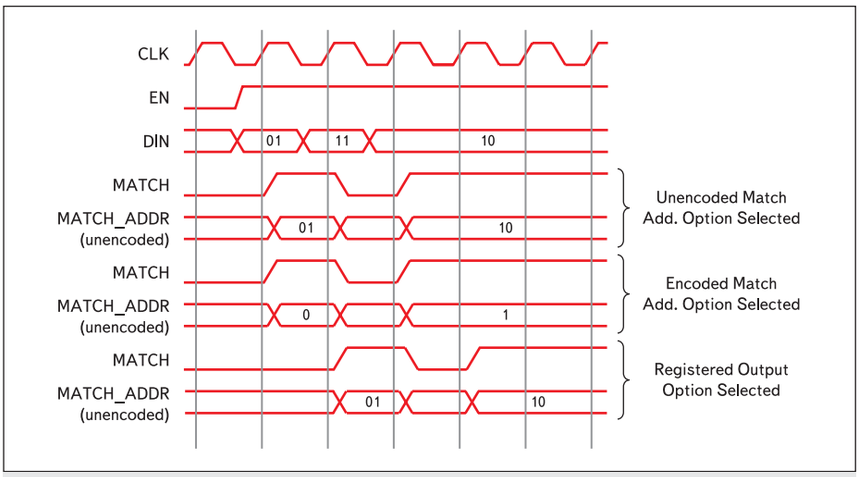

Рис. 129. Временные диаграммы, поясняющие выполнение операции чтения данных в контекстно-адресуемом запоминающем устройстве, создаваемом с помощью параметризированного модуля Content Addressable Memory версии v5.1 для последующей реализации на базе ресурсов блочной памяти ПЛИС Block SelectRAM

В том случае, если в нажатом состоянии находится кнопка *Single Match Unencoded* или *Multi Match Unencoded*, разрядность выходной шины адреса совпадает с количеством ячеек в создаваемом элементе памяти:

$$K = Depth. \qquad (34)$$

Каждый проводник (разряд) этой шины соответствует определенному адресу ячейки генерируемого контекстно-адресуемого запоминающего устройства. При нажатой кнопке *Single Match Unencoded* адрес ячейки, содержимое которой соответствует комбинации значений, представленных на входных шинах данных и маски этого элемента ассоциативной памяти, отображается в виде активного уровня сигнала только на одном из адресных выходов. В случае обнаружения нескольких ячеек, содержащих информацию, соответствующую комбинации кодов входных данных и маски, выбор значения отображаемого адреса (самого младшего или самого старшего) определяется установленным значением параметра *Address Resolution*.

Чтобы сформировать описание элемента контекстно-адресуемого запоминающего устройства, позволяющего отображать адреса всех ячеек, содержимое которых удовлетворяет заданному условию совпадения, следует переключить в нажатое состояние кнопку *Multi Match Unencoded*. В этом случае активный уровень сигнала может одновременно присутствовать на нескольких линиях (проводниках) выходной адресной шины, которые соответствуют ячейкам, содержащим искомые данные.

Выбор требуемого варианта значения параметра *Address Resolution* осуществляется с помощью двух кнопок с зависимой фиксацией, которые находятся во встроенной панели *Additional Features* (рис. 128). Для подготовки описания элемента ассоциативной памяти, формирующего на выходной шине значение самого младшего адреса ячейки, содержащей требуемые данные, следует нажать кнопку *Lowest*. Если в генерируемом элементе контекстно-адресуемого запоминающего устройства на выходной шине должно отображаться значение старшего адреса ячейки, содержимое которой совпадает с комбинацией значений входных данных и маски, то нужно переключить в нажатое состояние кнопку *Highest*.

На рис. 129 приведены временные диаграммы, которые поясняют последовательность выполнения операции чтения данных в контекстно-адресуемом запоминающем устройстве, реализуемом на базе ресурсов блочной памяти ПЛИС Block SelectRAM, при различных вариантах представления адреса на выходной шине (различных значениях параметра *Match Address Type*). На этом же рисунке показаны временные диаграммы для операции чтения данных при использовании выходных регистров в создаваемом элементе.

Временные диаграммы, демонстрирующие последовательность выполнения операции чтения данных в элементе ассоциативной памяти, реализуемом на основе сдвиговых регистров SRL16, при различных вариантах значения параметра *Match Address Type* показаны на рис. 130.

Представленные временные диаграммы соответствуют режиму раздельного выполнения операций чтения и записи данных в контекстно-адресуемом запоминающем устройстве. При этом входная шина данных DIN используется как для чтения, так и для записи входных данных. Чтобы сгенерировать описание элемента ассоциативной памяти, функционирующего в режиме одновременного выполнения операций записи и чтения данных, нужно установить в состояние «Включено» индикатор *Simultaneous Read Write*, который расположен во встроенной па-

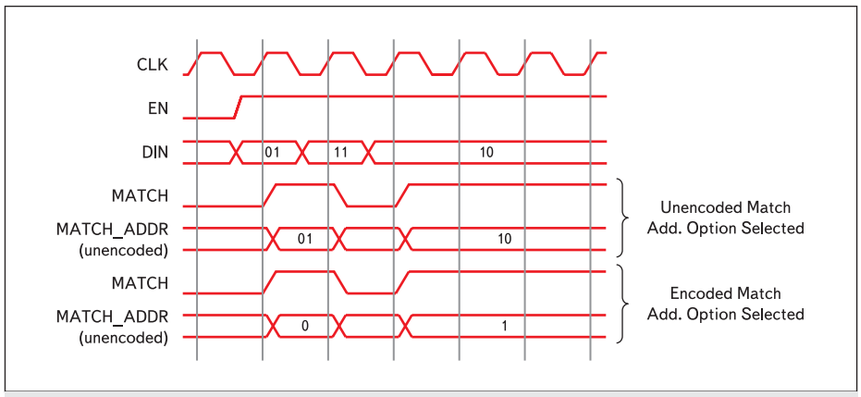

Рис. 130. Временные диаграммы, поясняющие выполнение операции чтения данных в контекстно-адресуемом запоминающем устройстве, создаваемом с помощью параметризированного модуля Content Addressable Memory версии v5.1 для последующей реализации на основе сдвиговых регистров SRL16

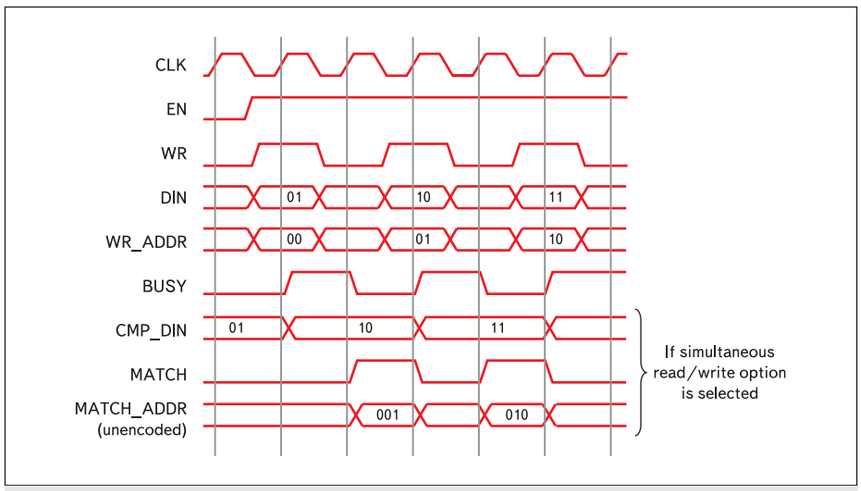

Рис. 131. Временные диаграммы, поясняющие одновременное выполнение операций записи и чтения данных в контекстно-адресуемом запоминающем устройстве, создаваемом с помощью параметризированного модуля Content Addressable Memory версии v5.1 для последующей реализации на базе ресурсов блочной памяти ПЛИС Block SelectRAM

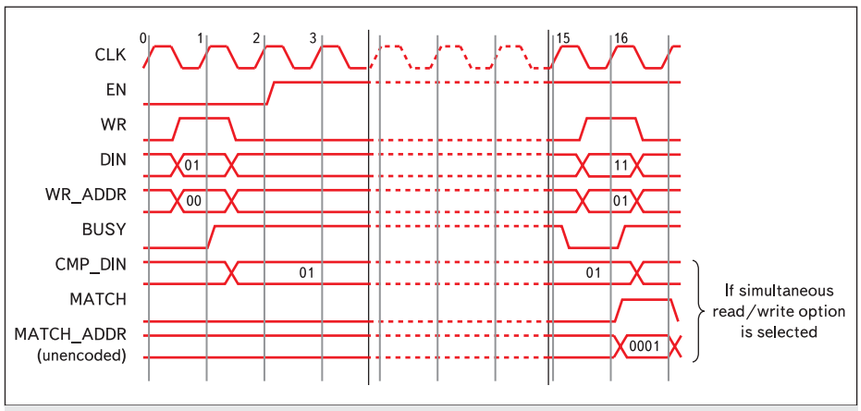

Рис. 132. Временные диаграммы, демонстрирующие одновременное выполнение операций записи и чтения данных в контекстно-адресуемом запоминающем устройстве, создаваемом с помощью параметризированного модуля Content Addressable Memory версии v5.1 для последующей реализации на основе сдвиговых регистров SRL16

нели *Additional Features* (рис. 128). В этом случае в состав интерфейса создаваемого элемента добавляется дополнительная шина данных CMP_DIN, которая предназначена только для операции чтения (поиска) требуемого значения. Записываемая информация при этом поступает на входную шину данных DIN и заносится в ячейку ассоциативной памяти, адрес которой представлен на входной адресной шине. На рис. 131 изображены временные диаграммы, которые в наглядной форме поясняют одновременное выполнение операций записи и чтения данных в контекстно-адресуемом запоминающем устройстве, реализуемом на базе ресурсов блочной памяти ПЛИС Block SelectRAM.

Временные диаграммы, соответствующие режиму одновременного выполнения операций записи и чтения данных в элементе ассоциативной памяти, реализуемом на основе сдвиговых регистров SRL16, представлены на рис. 132.

Параметризированный модуль Content Addressable Memory версии v5.1 позволяет формировать описания элементов контекстно-адресуемых запоминающих устройств, реализуемых на базе сдвиговых регистров SRL16, в которых для чтения (поиска) необходимых данных дополнительно используется соответствующая маска. При этом разработчик может выбрать один из двух режимов маскирования: стандартный или расширенный. При использовании стандартного режима маскирования разряды слова данных, определяющего код поиска, могут принимать одно из следующих трех значений: значение логического нуля (0), значение логической единицы (1) и безразличное состояние (X). Расширенный режим маскирования допускает присутствие в составе входного слова данных разрядов, состояние которых не анализируется (не принимается во внимание) при выполнении операции чтения. Такие разряды обозначаются символом U. Для формирования необходимой маски в состав интерфейса генерируемого элемента добавляется соответствующая входная шина DATA_MASK, которая имеет ту же разрядность, что и входная шина данных. Если в создаваемом элементе ассоциативной памяти применяется режим одновременного выполнения операций записи и чтения данных, то код маски поступает на входную шину CMP_DATA_MASK, которая автоматически добавляется в состав интерфейса этого элемента.

Интерпретация значений разрядов кода маски зависит от используемого режима маскирования. Если выбран стандартный режим маскирования, то единичное значение какого-либо разряда кода маски преобразует в безразличное состояние (X) значение аналогичного разряда входного слова данных, представленного на шине DIN (или CMP_DIN при использовании режима одновременного выполнения операций чтения и записи данных). При этом нулевое значение разряда кода маски оставляет исходное значение соответствующего разряда входного слова данных без изменения. Когда используется расширенный режим маскирования, значение каждого разряда искомого слова данных определяется комбинацией значений соответствую-

щих разрядов кода маски и входного слова данных, представленного на шине DIN (или CMP_DIN). Если аналогичные разряды кода маски и входного слова данных принимают одновременно нулевое значение, то при выполнении операции чтения (поиска) значения этих разрядов воспринимаются как безразличное состояние (X). Одновременное присутствие единичного значения в соответствующих разрядах слова данных и кода маски указывает на то, что состояние данного разряда слова данных не будет анализироваться при выполнении операции чтения. Когда некоторый разряд слова данных принимает значение логической единицы, а аналогичный разряд кода маски — значение логического нуля, то состояние данного разряда при осуществлении операции чтения воспринимается как единичное значение. Противоположная комбинация значений соответствующих разрядов слова данных и кода маски интерпретируется как нулевое значение.

Для выбора режима маскирования в создаваемом элементе предназначена группа кнопок с зависимой фиксацией, которая представлена во встроенной панели *Ternary Mode* (рис. 128). При нажатой кнопке *Ternary Mode Off* формируется описание контекстно-адресуемого запоминающего устройства, в котором возможность маскирования входных данных отсутствует. Для генерации элемента ассоциативной памяти, поддерживающего стандартный режим маскирования, необходимо переключить в нажатое состояние кнопку *Standard Ternary Mode*. Чтобы сформировать описание элемента контекстно-адресуемой памяти, использующего расширенный режим маскирования, нужно нажать кнопку *Enhanced Ternary Mode*.

При подготовке описаний элементов ассоциативных запоминающих устройств, реализуемых на основе ресурсов блочной памяти ПЛИС Block SelectRAM, в их состав могут быть включены выходные регистры. Для этой цели предназначен индикатор состояния *Register Outputs*, расположенный во встроенной панели *Additional Features* (рис. 128). Если данный индикатор находится в состоянии «Включено», то выходные регистры устанавливаются в цепях сигналов MATCH_ADDR, MATCH, SINGLE_MATCH, MULTIPLE_MATCH. При этом следует учитывать, что применение выходных регистров вносит дополнительную задержку при выполнении операции чтения, значение которой равно одному периоду тактового сигнала.

Вторая, заключительная, диалоговая панель «мастера» настройки параметров генератора оперативных и постоянных контекстно-адресуемых запоминающих устройств Content Addressable Memory версии v5.1 предназначена для выбора дополнительных входов сигналов управления и информационных выходов, а также для определения файла инициализации содержимого создаваемого элемента ассоциативной памяти. В этой же

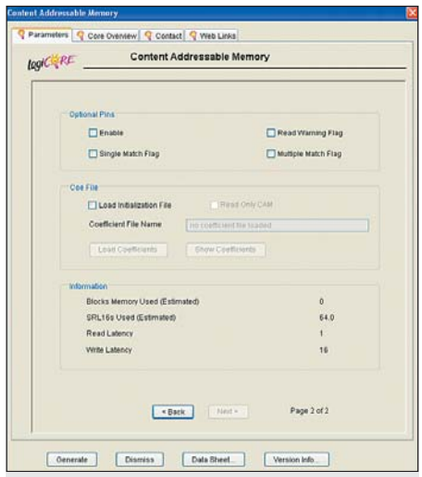

Рис. 133. Вид заключительной диалоговой панели «мастера» настройки параметров генератора элементов ассоциативной памяти Content Addressable Memory версии v5.1

диалоговой панели отображается оценочная информация о количестве используемых ресурсов кристалла для реализации генерируемого элемента и значениях задержек при выполнении операций чтения и записи данных. Вид заключительной диалоговой панели «мастера» настройки параметров рассматриваемого ядра представлен на рис. 133.

Для выбора дополнительных входов сигналов управления и информационных выходов, включаемых в состав интерфейса формируемого элемента ассоциативной памяти, следует воспользоваться индикаторами состояния, которые представлены во встроенной панели *Optional Pins* (рис. 133). Чтобы задействовать в генерируемом элементе контекстно-адресуемого запоминающего устройства вход разрешения/запрета выполнения операций чтения и записи, следует установить индикатор *Enable* в состояние «Включено». В этом случае все операции в формируемом элементе будут производиться только при подаче активного уровня сигнала на вход разрешения EN. Индикатор *Single Match Flag* позволяет добавить в состав интерфейса создаваемого элемента выход сигнала SINGLE_MATCH, информирующего о наличии только одной ячейки памяти, содержимое которой соответствует комбинации кодов, представленных на входных шинах данных и маски. При отсутствии таких ячеек или наличии более одной ячейки выход данного сигнала (флаг) находится в неактивном (сброшенном) состоянии. С помощью индикатора *Multiple Match Flag* пользователь может включить в состав интерфейса формируемого элемента выход сигнала MULTIPLE_MATCH, активный уровень которого сообщает о наличии нескольких ячеек ассоциативной памяти, содержимое которых соответствует комбинации кодов, представленных на входных шинах данных и маски. Если найдена только одна ячейка или ни одной, то данный сигнал (флаг) остается в сброшенном (неактивном) состоянии. В элементах контекстно-адресуемых запоминающих устройств, использующих режим одновременного выполнения чтения и записи данных, целесообразно задействовать выход сигнала предупреждения RD_WARNING. Активный уровень этого сигнала указывает на то, что текущие данные, предназначенные для записи в этот элемент, совпадают со значением, представленным на входной шине для чтения данных. Так как для выполнения операции записи данных необходимо значительно большее время, чем для операции чтения, то информация, отображаемая на выходной адресной шине в этом случае, может не соответствовать реальной ситуации (носить недостоверный характер). Чтобы использовать в создаваемом элементе контекстно-адресуемой памяти выход сигнала предупреждения RD_WARNING, следует перевести в состояние «Включено» индикатор *Read Warning Flag*.

Для автоматической инициализации содержимого генерируемых элементов ассоциативных запоминающих устройств следует определить значения соответствующих параметров, воспользовавшись элементами управления, которые расположены во встроенной панели *COE File* (рис. 133). Информация, которая должна быть автоматически загружена в соответствующие ячейки формируемого элемента контекстно-адресуемой памяти, указывается в виде файла формата COE. Чтобы определить название соответствующего файла на диске, содержащего требуемые данные, необходимо, прежде всего, переключить индикатор *Load Initialization File* в состояние «Включено». После этого становятся доступными поле редактирования *Coefficient File Name* и кнопка *Load Coefficients*. При нажатии на данную кнопку на экран выводится стандартная панель диалога открытия файла, которая позволяет быстро найти на одном из дисков компьютера требуемый файл, описывающий содержимое создаваемого ассоциативного запоминающего устройства. После выбора соответствующего файла и закрытия стандартной диалоговой панели его название автоматически отображается в поле редактирования *Coefficient File Name*. Можно также с помощью клавиатуры сразу указать в этом поле редактирования название требуемого файла инициализации, не выполняя процедуру его поиска. Для быстрого просмотра содержимого выбранного файла инициализации можно воспользоваться кнопкой *Show Coefficients*, которая также находится во встроенной панели *COE File*. Наиболее эффективные способы подготовки файлов, которые предназначены для инициализации содержимого элементов запоминающих устройств, будут рассмотрены в следующей части настоящей статьи.

Чтобы сформировать описание элемента постоянного контекстно-адресуемого запоминающего устройства (функционирующего

в режиме только чтения данных), следует перевести в состояние «Включено» индикатор *Read Only CAM*. Для этого нужно вначале определить название файла, в котором находится описание содержимого генерируемого элемента ассоциативной памяти. Только после этого становится доступным индикатор состояния *Read Only CAM*.

Предварительная оценочная информация о количестве ресурсов кристалла, используемых для реализации генерируемого элемента памяти, отображается в краткой форме во встроенной панели *Information* (рис. 133). В строке *Blocks Memory Used (Estimated)* приводятся данные о количестве модулей блочной памяти ПЛИС Block SelectRAM, которое потребуется для создания элемента ассоциативного запоминающего устройства с указанными параметрами конфигурации. В строке *SRL16s Used (Estimated)* отображается число сдвиговых регистров SRL16, которое необходимо для реализации формируемого элемента контекстно-адресуемой памяти. Более подробная и достоверная информация об объеме различных ресурсов ПЛИС, необходимых для реализации формируемого элемента ассоциативной памяти, может быть получена только после завершения процесса генерации этого элемента.

В этой же встроенной панели приводятся сведения о суммарной длительности задержки при выполнении операций чтения и записи данных, выраженной в количестве периодов тактового сигнала. Значение задержки появления адреса ячейки, содержащей искомые данные, на выходной шине адреса при выполнении операции чтения представлено в строке *Read Latency*. Значение задержки записи данных в формируемом элементе отображается в строке *Write Latency*.

Если какие-либо сведения, представленные во встроенной панели *Information*, не соответствуют возможностям выбранного кристалла, то необходимо скорректировать значение соответствующих параметров создаваемого элемента ассоциативной памяти, вернувшись с помощью кнопки *Назад (Back)* к предыдущей диалоговой панели «мастера» настройки параметров ядра Content Addressable Memory версии v5.1. Если при этом не удается достигнуть приемлемых результатов, то следует выбрать другой тип ПЛИС, возможности которого соответствуют параметрам генерируемого элемента постоянного контекстно-адресуемого запоминающего устройства.

В элементах ассоциативных запоминающих устройств, создаваемых с помощью рассматриваемого параметризированного модуля, используется следующая система условных обозначений входных и выходных портов:

- clk — вход сигнала синхронизации;
- cmp_data_mask[N:0] — входная шина с разрядностью N+1, на которую подается код маски, в элементах, использующих режим одновременного выполнения операций чтения и записи данных;
- cmp_din[N:0] — входная шина с разрядностью N+1, на которую поступает код считываемых данных, в элементах, использующих режим одновременного выполнения операций чтения и записи данных;
- data_mask[N:0] — входная шина с разрядностью N+1, на которую подается код маски, в элементах, использующих режим раздельного выполнения операций чтения и записи данных;
- din[N:0] — входная шина с разрядностью N+1, на которую поступает код считываемых данных, в элементах, использующих режим раздельного выполнения операций чтения и записи данных, или записываемая информация в элементах, использующих режим одновременного выполнения операций чтения и записи данных;
- en — вход сигнала разрешения выполнения операций чтения и записи данных;
- we — вход сигнала записи данных;
- wr_addr[M:0] — входная шина с разрядностью M+1, на которую подается адрес записываемых данных;
- busy — выход сигнала, информирующего о ходе выполнения текущей операции записи данных;
- match — выход сигнала, сообщающего о наличии хотя бы одной ячейки, содержащей данные, которые соответствуют комбинации значений, представленных на входных шинах данных и маски;
- match_addr[M:0] — выходная шина с разрядностью M+1, на которой отображается адрес ячейки, содержащей информацию, которая соответствует комбинации кодов, представленных на входных шинах данных и маски;
- multiple_match — выход сигнала, информирующего о наличии нескольких ячеек, содержимое которых удовлетворяет условию поиска (чтения) данных;
- read_warning — выход сигнала, предупреждающего о том, что данные, записываемые в текущем цикле в этот элемент, совпадают с кодом, представленным на входной шине для чтения данных;
- single_match — выход сигнала, сообщающего о наличии только одной ячейки, содержащей данные, которые соответствуют комбинации значений, представленных на входных шинах данных и маски.

Пример описания элемента ассоциативного запоминающего устройства, реализуемого на основе сдвиговых регистров SRL16, сформированного с помощью ядра Content Addressable Memory версии v5.1 средствами генератора параметризированных модулей CORE Generator

В качестве примера элемента оперативной ассоциативной памяти, созданного с помощью ядра Content Addressable Memory версии v5.1 для последующей реализации на основе сдвиговых регистров SRL16, приводится VHDL-описание контекстно-адресуемого запоминающего устройства *cam_v5_1_srl16e_1024_16* с информационной емкостью 16 кбит и организацией 1 Кслов × 16 разрядов. Данный элемент поддерживает режим одновременного выполнения операций записи и чтения данных. Для поиска требуемой информации в элементе *cam_v5_1_srl16e_1024_16* может использоваться расширенный режим маскирования. При этом адрес ячейки, содержимое которой соответствует комбинации кодов, представленных на входных шинах данных и маски, отображается на выходной шине адреса формируемого элемента в виде десятиразрядного двоичного кода. В случае обнаружения нескольких ячеек, содержимое которых удовлетворяет заданным условиям поиска данных, на выходной адресной шине будет представлено значение самого младшего адреса ячейки:

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
-- synthesis translate_off
Library XilinxCoreLib;
-- synthesis translate_on
ENTITY cam_v5_1_srl16e_1024_16 IS
   port (
         clk: IN std_logic;
         cmp_data_mask: IN std_logic_VECTOR(15 downto 0);
         cmp_din: IN std_logic_VECTOR(15 downto 0);
         data_mask: IN std_logic_VECTOR(15 downto 0);
         din: IN std_logic_VECTOR(15 downto 0);
         en: IN std_logic;
         we: IN std_logic;
         wr_addr: IN std_logic_VECTOR(9 downto 0);
         busy: OUT std_logic;
         match: OUT std_logic;
         match_addr: OUT std_logic_VECTOR(9 downto 0);
         multiple_match: OUT std_logic;
         read_warning: OUT std_logic;
         single_match: OUT std_logic
         );
END cam_v5_1_srl16e_1024_16;
--
ARCHITECTURE cam_v5_1_srl16e_1024_16_a OF cam_v5_1_srl16e_
1024_16 IS
-- synthesis translate_off
component wrapped_cam_v5_1_srl16e_1024_16
   port (
         clk: IN std_logic;
         cmp_data_mask: IN std_logic_VECTOR(15 downto 0);
         cmp_din: IN std_logic_VECTOR(15 downto 0);
         data_mask: IN std_logic_VECTOR(15 downto 0);
         din: IN std_logic_VECTOR(15 downto 0);
         en: IN std_logic;
         we: IN std_logic;
         wr_addr: IN std_logic_VECTOR(9 downto 0);
         busy: OUT std_logic;
         match: OUT std_logic;
         match_addr: OUT std_logic_VECTOR(9 downto 0);
         multiple_match: OUT std_logic;
         read_warning: OUT std_logic;
         single_match: OUT std_logic
         );
end component;
--
-- Configuration specification
   for all : wrapped_cam_v5_1_srl16e_1024_16 use entity
XilinxCoreLib.cam_v5_1(behavioral)
         generic map(
                  c_has_en => 1,
                  c_wr_addr_width => 10,
                  c_din_width => 16,
                  c_has_wr_addr => 1,
                  c_data_mask_width => 16,
                  c_cmp_din_width => 16,
                  c_has_read_warning => 1,
                  c_width => 16,
                  c_mem_init => 0,
                  c_has_cmp_data_mask => 1,
                  c_has_we => 1,
```

```
                    c_enable_rlocs => 0,
                    c_addr_type => 0,
                    c_ternary_mode => 2,
                    c_match_resolution_type => 0,
                    c_has_single_match => 1,
                    c_depth => 1024,
                    c_has_multiple_match => 1,
                    c_read_cycles => 1,
                    c_mem_type => 0,
                    c_has_data_mask => 1,
                    c_reg_outputs => 0,
                    c_has_cmp_din => 1,
                    c_mem_init_file => «cam_v5_1_srl16e_1024_16.mif»,
                    c_cmp_data_mask_width => 16,
                    c_match_addr_width => 10
                    );
-- synthesis translate_on
BEGIN
-- synthesis translate_off
U0 : wrapped_cam_v5_1_srl16e_1024_16
        port map (
                    clk => clk,
                    cmp_data_mask => cmp_data_mask,
                    cmp_din => cmp_din,
                    data_mask => data_mask,
                    din => din,
                    en => en,
                    we => we,
                    wr_addr => wr_addr,
                    busy => busy,
                    match => match,
                    match_addr => match_addr,
                    multiple_match => multiple_match,
                    read_warning => read_warning,
                    single_match => single_match
                    );
-- synthesis translate_on
--
END cam_v5_1_srl16e_1024_16_a;
```

Операции записи и чтения информации в элементе *cam_v5_1_srl16e_1024_16* осуществляются только при наличии активного уровня сигнала на соответствующем входе разрешения EN. В состав интерфейса рассматриваемого элемента включены дополнительные выходы сигналов, информирующих о наличии одной или нескольких ячеек, содержимое которых удовлетворяет условию совпадения, определяемому совокупностью кодов, представленных на входных шинах данных и маски.

Для использования элемента ассоциативной памяти *cam_v5_1_srl16e_1024_16* в качестве одного из компонентов в составе VHDL-описания проектируемого устройства необходимо включить в раздел декларации этого описания следующую конструкцию:

```
component cam_v5_1_srl16e_1024_16
   port (
         clk: IN std_logic;
         cmp_data_mask: IN std_logic_VECTOR(15 downto 0);
         cmp_din: IN std_logic_VECTOR(15 downto 0);
         data_mask: IN std_logic_VECTOR(15 downto 0);
         din: IN std_logic_VECTOR(15 downto 0);
         en: IN std_logic;
         we: IN std_logic;
         wr_addr: IN std_logic_VECTOR(9 downto 0);
         busy: OUT std_logic;
         match: OUT std_logic;
         match_addr: OUT std_logic_VECTOR(9 downto 0);
         multiple_match: OUT std_logic;
         read_warning: OUT std_logic;
         single_match: OUT std_logic
);
end component;
--
-- Synplicity black box declaration
attribute syn_black_box : boolean;
attribute syn_black_box of cam_v5_1_srl16e_1024_16: component is true;
```

Создание конкретного экземпляра компонента контекстно-адресуемого запоминающего устройства *cam_v5_1_srl16e_1024_16* в описании разрабатываемого устройства выполняется с помощью оператора, шаблон которого имеет следующий вид:

```
<идентификатор_экземпляра_компонента_cam_v5_1_srl16e_1024_
16>: cam_v5_1_srl16e_1024_16
        port map (
                    clk => clk,
                    cmp_data_mask => cmp_data_mask,
                    cmp_din => cmp_din,
                    data_mask => data_mask,
                    din => din,
                    en => en,
                    we => we,
                    wr_addr => wr_addr,
                    busy => busy,
                    match => match,
                    match_addr => match_addr,
                    multiple_match => multiple_match,
                    read_warning => read_warning,
                    single_match => single_match
                    );
```

Подробные сведения о количестве таблиц преобразования LookUp Table (LUT), секций Slices и триггеров Flip Flop, используемых для автономной реализации элемента ассоциативной памяти *cam_v5_1_srl16e_1024_16*, представлены в информационной панели, вид которой показан на рис. 134.

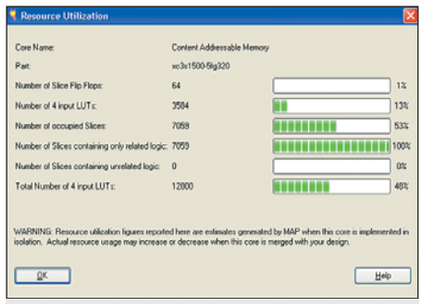

Рис. 134. Вид информационной панели, содержащей сведения о различных ресурсах ПЛИС, используемых для реализации элемента оперативной контекстно-адресуемой памяти cam_v5_1_srl16e_1024_16

Пример описания элемента контекстно-адресуемого запоминающего устройства, реализуемого на основе блочной памяти ПЛИС, сформированного с помощью ядра Content Addressable Memory версии v5.1 средствами генератора параметризированных модулей CORE Generator

Примером элемента оперативной ассоциативной памяти, реализуемого на основе блочной памяти ПЛИС и сгенерированного с помощью параметризированного модуля Content Addressable Memory версии v5.1, является контекстно-адресуемое запоминающее устройство *cam_v5_1_block_memory_512_18*, текст описания которого выглядит следующим образом:

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
-- synthesis translate_off
Library XilinxCoreLib;
-- synthesis translate_on
ENTITY cam_v5_1_block_memory_512_18 IS
   port (
         clk: IN std_logic;
         cmp_din: IN std_logic_VECTOR(17 downto 0);
         din: IN std_logic_VECTOR(17 downto 0);
         en: IN std_logic;
         we: IN std_logic;
         wr_addr: IN std_logic_VECTOR(8 downto 0);
         busy: OUT std_logic;
         match: OUT std_logic;
         match_addr: OUT std_logic_VECTOR(8 downto 0);
         multiple_match: OUT std_logic;
         read_warning: OUT std_logic;
         single_match: OUT std_logic
         );
END cam_v5_1_block_memory_512_18;
--
ARCHITECTURE cam_v5_1_block_memory_512_18_a OF
cam_v5_1_block_memory_512_18 IS
-- synthesis translate_off
component wrapped_cam_v5_1_block_memory_512_18
   port (
         clk: IN std_logic;
         cmp_din: IN std_logic_VECTOR(17 downto 0);
         din: IN std_logic_VECTOR(17 downto 0);
         en: IN std_logic;
         we: IN std_logic;
         wr_addr: IN std_logic_VECTOR(8 downto 0);
         busy: OUT std_logic;
         match: OUT std_logic;
         match_addr: OUT std_logic_VECTOR(8 downto 0);
         multiple_match: OUT std_logic;
         read_warning: OUT std_logic;
         single_match: OUT std_logic
         );
end component;
--
-- Configuration specification
   for all : wrapped_cam_v5_1_block_memory_512_18 use entity
XilinxCoreLib.cam_v5_1(behavioral)
         generic map(
                    c_has_en => 1,
                    c_wr_addr_width => 9,
                    c_din_width => 18,
                    c_has_wr_addr => 1,
                    c_data_mask_width => 18,
                    c_cmp_din_width => 18,
                    c_has_read_warning => 1,
                    c_width => 18,
                    c_mem_init => 0,
                    c_has_cmp_data_mask => 0,
                    c_has_we => 1,
                    c_enable_rlocs => 0,
                    c_addr_type => 0,
                    c_ternary_mode => 0,
                    c_match_resolution_type => 1,
                    c_has_single_match => 1,
                    c_depth => 512,
                    c_has_multiple_match => 1,
                    c_read_cycles => 1,
                    c_mem_type => 1,
                    c_has_data_mask => 0,
                    c_reg_outputs => 1,
                    c_has_cmp_din => 1,
                    c_mem_init_file => «cam_v5_1_block_memory_
512_18.mif»,
                    c_cmp_data_mask_width => 18,
                    c_match_addr_width => 9
                    );
-- synthesis translate_on
BEGIN
-- synthesis translate_off
U0 : wrapped_cam_v5_1_block_memory_512_18
         port map (
                    clk => clk,
                    cmp_din => cmp_din,
                    din => din,
                    en => en,
                    we => we,
                    wr_addr => wr_addr,
                    busy => busy,
                    match => match,
                    match_addr => match_addr,
                    multiple_match => multiple_match,
                    read_warning => read_warning,
                    single_match => single_match
                    );
-- synthesis translate_on
--
END cam_v5_1_block_memory_512_18_a;
```

Представленный элемент ассоциативной памяти *cam_v5_1_block_memory_512_18* имеет информационную емкость 9 кбит с организацией 512 слов × 18 разрядов. Разрядность входной и выходной адресной шины в этом элементе равна девяти. Адрес ячейки, содержащей информацию, которая соответствует значению, представленному на входной шине данных, отображается на выходной шине адреса сгенерированного элемента в виде 9-разрядного двоичного кода. При обнаружении нескольких ячеек, содержимое которых совпадает со входными данными, на выходной шине адреса отображается значение только самого старшего адреса. В состав архитектуры запоминающего устройства *cam_v5_1_block_memory_512_18* с целью повышения производительности включены выходные регистры. В этом элементе применяется также режим одновременного выполнения операций записи и чтения данных. В состав интерфейса элемента *cam_v5_1_block_memory_512_18* добавлены дополнительные выходы сигналов, информирующих о количестве обнаруженных ячеек, содержимое которых совпадает со значением, представленным на входной шине данных. Выполнение операций записи и чтения данных в рассматриваемом элементе производится только при подаче активного уровня сигнала на соответствующий вход разрешения EN.

Декларация компонента *cam_v5_1_block_memory_512_18* в описании разрабатываемого устройства осуществляется с помощью приведенной последовательности выражений:

```
component cam_v5_1_block_memory_512_18
   port (
         clk: IN std_logic;
         cmp_din: IN std_logic_VECTOR(17 downto 0);
         din: IN std_logic_VECTOR(17 downto 0);
         en: IN std_logic;
         we: IN std_logic;
         wr_addr: IN std_logic_VECTOR(8 downto 0);
         busy: OUT std_logic;
         match: OUT std_logic;
         match_addr: OUT std_logic_VECTOR(8 downto 0);
         multiple_match: OUT std_logic;
         read_warning: OUT std_logic;
         single_match: OUT std_logic
         );
end component;
--
-- Synplicity black box declaration
attribute syn_black_box : boolean;
attribute syn_black_box of cam_v5_1_block_memory_512_18: component is true;
```

Для описания конкретного экземпляра компонента *cam_v5_1_block_memory_512_18* необходимо добавить в состав блока определения архитектуры проектируемого устройства оператор, шаблон которого имеет следующий вид:

```
<идентификатор_экземпляра_элемента_cam_v5_1_block_memory_512_18>: cam_v5_1_block_memory_512_18
         port map (
                  clk => clk,
                  cmp_din => cmp_din,
                  din => din,
                  en => en,
                  we => we,
                  wr_addr => wr_addr,
                  busy => busy,
                  match => match,
                  match_addr => match_addr,
                  multiple_match => multiple_match,
                  read_warning => read_warning,
                  single_match => single_match
                  );
```

На рис. 135 приведена информационная панель, в которой содержатся детализированные сведения об объеме различных ресурсов ПЛИС (включая модули блочной памяти кристалла), используемых для автономной реализации элемента *cam_v5_1_block_memory_512_18*.

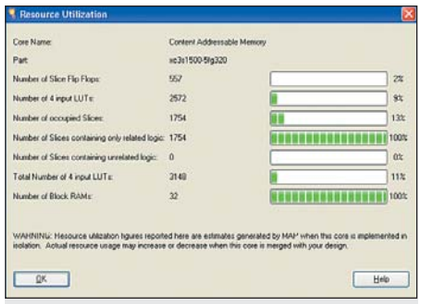

Рис. 135. Вид информационной панели, содержащей сведения о ресурсах ПЛИС, используемых для реализации элемента контекстно-адресуемого запоминающего устройства cam_v5_1_block_memory_512_18

Пример описания элемента постоянного ассоциативного запоминающего устройства, реализуемого на основе сдвиговых регистров SRL16, сформированного с помощью ядра Content Addressable Memory версии v5.1 средствами генератора параметризированных модулей CORE Generator

Формирование описаний элементов постоянной ассоциативной памяти, реализуемой на основе сдвиговых регистров SRL16, с помощью параметризированного модуля Content Addressable Memory версии v5.1 демонстрируется в настоящем разделе на примере контекстно-адресуемого запоминающего устройства *rom_cam_v5_1_srl16e_256_24*. Информационная емкость данного элемента ассоциативной памяти составляет 6 кбит с организацией 256 слов × 24 разряда. При чтении (поиске) требуемой информации в элементе *rom_cam_v5_1_srl16e_256_24* поддерживается стандартный режим маскирования. Представление адресов ячеек в декодированном виде, используемое в этом элементе ассоциативной памяти, позволяет одновременно отображать информацию обо всех ячейках запоминающего устройства, содержащих требуемые данные. Адреса ячеек, содержимое которых соответствует комбинации кодов, присутствующих на входных шинах данных и маски, отображаются в виде высокого уровня сигнала на соответствующих линиях выходной адресной шины. В составе интерфейса элемента *rom_cam_v5_1_srl16e_256_24* присутствуют те же дополнительные выходы информационных сигналов, что и в элементе оперативной контекстно-адресуемой памяти *cam_v5_1_block_memory_512_18*, представленном в предыдущем разделе:

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
-- synthesis translate_off
Library XilinxCoreLib;
-- synthesis translate_on
ENTITY rom_cam_v5_1_srl16e_256_24 IS
   port (
         clk: IN std_logic;
         data_mask: IN std_logic_VECTOR(23 downto 0);
         din: IN std_logic_VECTOR(23 downto 0);
         en: IN std_logic;
         busy: OUT std_logic;
         match: OUT std_logic;
         match_addr: OUT std_logic_VECTOR(255 downto 0);
         multiple_match: OUT std_logic;
         read_warning: OUT std_logic;
         single_match: OUT std_logic
         );
END rom_cam_v5_1_srl16e_256_24;
--
ARCHITECTURE rom_cam_v5_1_srl16e_256_24_a OF rom_cam_v5_1_srl16e_256_24 IS
-- synthesis translate_off
component wrapped_rom_cam_v5_1_srl16e_256_24
   port (
         clk: IN std_logic;
         data_mask: IN std_logic_VECTOR(23 downto 0);
         din: IN std_logic_VECTOR(23 downto 0);
         en: IN std_logic;
         busy: OUT std_logic;
         match: OUT std_logic;
         match_addr: OUT std_logic_VECTOR(255 downto 0);
         multiple_match: OUT std_logic;
         read_warning: OUT std_logic;
         single_match: OUT std_logic
         );
end component;
--
-- Configuration specification
   for all : wrapped_rom_cam_v5_1_srl16e_256_24 use entity XilinxCoreLib.cam_v5_1(behavioral)
         generic map(
                  c_has_en => 1,
                  c_wr_addr_width => 8,
                  c_din_width => 24,
                  c_has_wr_addr => 0,
                  c_data_mask_width => 24,
                  c_cmp_din_width => 24,
                  c_has_read_warning => 1,
                  c_width => 24,
                  c_mem_init => 1,
                  c_has_cmp_data_mask => 0,
                  c_has_we => 0,
                  c_enable_rlocs => 0,
                  c_addr_type => 2,
                  c_ternary_mode => 1,
                  c_match_resolution_type => 0,
                  c_has_single_match => 1,
                  c_depth => 256,
                  c_has_multiple_match => 1,
                  c_read_cycles => 1,
                  c_mem_type => 0,
                  c_has_data_mask => 1,
                  c_reg_outputs => 0,
                  c_has_cmp_din => 0,
                  c_mem_init_file => «rom_cam_v5_1_srl16e_256_24.mif»,
                  c_cmp_data_mask_width => 24,
                  c_match_addr_width => 256
                  );
-- synthesis translate_on
BEGIN
-- synthesis translate_off
U0 : wrapped_rom_cam_v5_1_srl16e_256_24
         port map (
                  clk => clk,
                  data_mask => data_mask,
                  din => din,
                  en => en,
                  busy => busy,
                  match => match,
                  match_addr => match_addr,
```

```
                multiple_match => multiple_match,
                read_warning => read_warning,
                single_match => single_match
                );
-- synthesis translate_on
--
END rom_cam_v5_1_srl16e_256_24_a;
```

Информация, используемая для инициализации содержимого ячеек сформированного элемента постоянного контекстно-адресуемого запоминающего устройства, указана в виде файла *rom_cam_v5_1_srl16e_256_24.coe*.

Блок декларации представленного выше элемента постоянной ассоциативной памяти *rom_cam_v5_1_srl16e_256_24* при использовании его в качестве компонента проектируемого устройства имеет следующий вид:

```
component rom_cam_v5_1_srl16e_256_24
  port (
        clk: IN std_logic;
        data_mask: IN std_logic_VECTOR(23 downto 0);
        din: IN std_logic_VECTOR(23 downto 0);
        en: IN std_logic;
        busy: OUT std_logic;
        match: OUT std_logic;
        match_addr: OUT std_logic_VECTOR(255 downto 0);
        multiple_match: OUT std_logic;
        read_warning: OUT std_logic;
        single_match: OUT std_logic
        );
end component;
--
-- Synplicity black box declaration
attribute syn_black_box : boolean;
attribute syn_black_box of rom_cam_v5_1_srl16e_256_24: compo-
nent is true;
```

Создание конкретного экземпляра компонента *rom_cam_v5_1_srl16e_256_24* в описании разрабатываемого устройства осуществляется с помощью приведенного оператора, который должен быть включен в состав архитектурного блока:

```
<идентификатор_экземпляра_элемента_rom_cam_v5_1_srl16e_256_
24> : rom_cam_v5_1_srl16e_256_24
        port map (
                clk => clk,
                data_mask => data_mask,
                din => din,
                en => en,
                busy => busy,
                match => match,
                match_addr => match_addr,
                multiple_match => multiple_match,
                read_warning => read_warning,
                single_match => single_match
                );
```

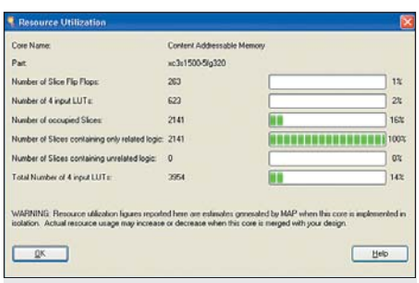

Рис. 136. Вид информационной панели, содержащей сведения о количестве различных ресурсов ПЛИС, используемых для реализации элемента постоянной ассоциативной памяти rom_cam_v5_1_srl16e_256_24

Исчерпывающие сведения о количестве различных ресурсов ПЛИС, которое необходимо для автономной реализации рассмотренного элемента постоянной ассоциативной памяти, отражены в информационной панели, вид которой представлен на рис. 136.

Пример описания элемента постоянного ассоциативного запоминающего устройства, реализуемого на базе ресурсов блочной памяти ПЛИС, сформированного с помощью ядра Content Addressable Memory версии v5.1 средствами генератора параметризированных модулей CORE Generator

Результат применения параметризированного модуля Content Addressable Memory версии v5.1 для создания постоянных контекстно-адресуемых запоминающих устройств, реализуемых на базе ресурсов блочной памяти ПЛИС Block SelectRAM, показан на примере описания элемента *rom_cam_v5_1_block_memory_384_16*, текст которого имеет следующий вид:

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
-- synthesis translate_off
Library XilinxCoreLib;
-- synthesis translate_on
ENTITY rom_cam_v5_1_block_memory_384_16 IS
  port (
        clk: IN std_logic;
        din: IN std_logic_VECTOR(15 downto 0);
        en: IN std_logic;
        busy: OUT std_logic;
        match: OUT std_logic;
        match_addr: OUT std_logic_VECTOR(383 downto 0);
        multiple_match: OUT std_logic;
        read_warning: OUT std_logic;
        single_match: OUT std_logic
        );
END rom_cam_v5_1_block_memory_384_16;
--
ARCHITECTURE rom_cam_v5_1_block_memory_384_16_a OF
rom_cam_v5_1_block_memory_384_16 IS
-- synthesis translate_off
component wrapped_rom_cam_v5_1_block_memory_384_16
  port (
        clk: IN std_logic;
        din: IN std_logic_VECTOR(15 downto 0);
        en: IN std_logic;
        busy: OUT std_logic;
        match: OUT std_logic;
        match_addr: OUT std_logic_VECTOR(383 downto 0);
        multiple_match: OUT std_logic;
        read_warning: OUT std_logic;
        single_match: OUT std_logic
        );
end component;
--
-- Configuration specification
  for all : wrapped_rom_cam_v5_1_block_memory_384_16 use en-
tity XilinxCoreLib.cam_v5_1(behavioral)
        generic map(
                c_has_en => 1,
                c_wr_addr_width => 9,
                c_din_width => 16,
                c_has_wr_addr => 0,
                c_data_mask_width => 16,
                c_cmp_din_width => 16,
                c_has_read_warning => 1,
                c_width => 16,
                c_mem_init => 1,
                c_has_cmp_data_mask => 0,
                c_has_we => 0,
                c_enable_rlocs => 0,
                c_addr_type => 1,
                c_ternary_mode => 0,
                c_match_resolution_type => 0,
                c_has_single_match => 1,
                c_depth => 384,
                c_has_multiple_match => 1,
                c_read_cycles => 1,
                c_mem_type => 1,
                c_has_data_mask => 0,
                c_reg_outputs => 1,
                c_has_cmp_din => 0,
                c_mem_init_file => «rom_cam_v5_1_block_
memory_384_16.mif»,
                c_cmp_data_mask_width => 16,
                c_match_addr_width => 384
                );


-- synthesis translate_on
BEGIN
-- synthesis translate_off
U0 : wrapped_rom_cam_v5_1_block_memory_384_16
        port map (
                clk => clk,
                din => din,
                en => en,
                busy => busy,
                match => match,
                match_addr => match_addr,
                multiple_match => multiple_match,
                read_warning => read_warning,
                single_match => single_match
                );
-- synthesis translate_on
--
END rom_cam_v5_1_block_memory_384_16_a;
```

Элемент постоянной ассоциативной памяти *rom_cam_v5_1_block_memory_384_16* имеет информационную емкость 6 кбит с организацией 384 слова × 16 разрядов. Адрес ячейки, содержимое которой совпадает со значением, представленным на входной шине данных, отображается в виде активного уровня сигнала на соответствующей линии выходной адресной шины. При наличии нескольких ячеек, содержащих требуемые данные, на выходную адресную шину выводится значение только самого младшего адреса. В состав структуры элемента *rom_cam_v5_1_block_memory_384_16* включены выходные регистры. Для автоматической инициализации содержимого ячеек сформированного элемента ассоциативной памяти используется файл с названием *rom_cam_v5_1_block_memory_384_16.coe*.

Декларация компонента *rom_cam_v5_1_block_memory_384_16* в составе VHDL-описания разрабатываемого устройства осуществляется с помощью приведенной последовательности выражений:

```
component rom_cam_v5_1_block_memory_384_16
  port (
        clk: IN std_logic;
        din: IN std_logic_VECTOR(15 downto 0);
        en: IN std_logic;
        busy: OUT std_logic;
        match: OUT std_logic;
        match_addr: OUT std_logic_VECTOR(383 downto 0);
        multiple_match: OUT std_logic;
        read_warning: OUT std_logic;
        single_match: OUT std_logic
        );
end component;
--
-- Synplicity black box declaration
attribute syn_black_box : boolean;
attribute syn_black_box of rom_cam_v5_1_block_memory_384_16:
component is true;
```

Для создания конкретного экземпляра компонента *rom_cam_v5_1_block_memory_384_16* нужно добавить в состав блока определения архитектуры проектируемого устройства оператор, шаблон которого имеет следующий вид:

```
<идентификатор_экземпляра_компонента_rom_cam_v5_1_block_
memory_384_16>: rom_cam_v5_1_block_memory_384_16
        port map (
                clk => clk,
```

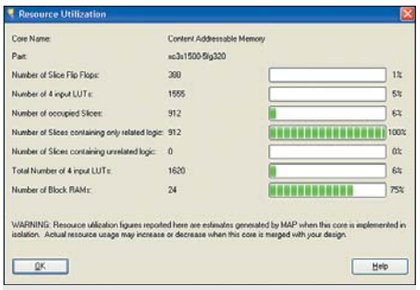

Рис. 137. Вид информационной панели, содержащей сведения о количестве различных ресурсов кристалла, используемых для реализации элемента постоянной ассоциативной памяти rom_cam_v5_1_block_memory_384_16

```
din => din,
en => en,
busy => busy,
match => match,
match_addr => match_addr,
multiple_match => multiple_match,
read_warning => read_warning,
single_match => single_match
);
```

Сведения о количестве триггеров Flip Flop, таблиц преобразования LUT, секций Slices и модулей блочной памяти ПЛИС Block SelectRAM, которое необходимо для автономной реализации элемента *rom_cam_v5_1_block_memory_384_16*, приведены в информационной панели, представленной на рис. 137.

Продолжение следует