2. Processor Architecture

NII51002-9.1.0

2. Архитектура процессора

Введение

В этой главе описывается аппаратная структура процессора Nios® II, обсуждаются все функциональные узлы архитектуры Nios II и основы аппаратной реализации процессора Nios II. Эта глава состоит из следующих секций:

■"Реализация процессора" на странице 2-2
■"Регистровый файл" на странице 2-3
■"Арифметико-логическое устройство" на странице 2-3
■"Сигналы сброса и отладки" на странице 2-7
■"Контроллеры исключений и прерываний" на странице 2-7
■"Организация памяти и I/O" на странице 2-10
■"Отладочный модуль JTAG" на странице 2-17

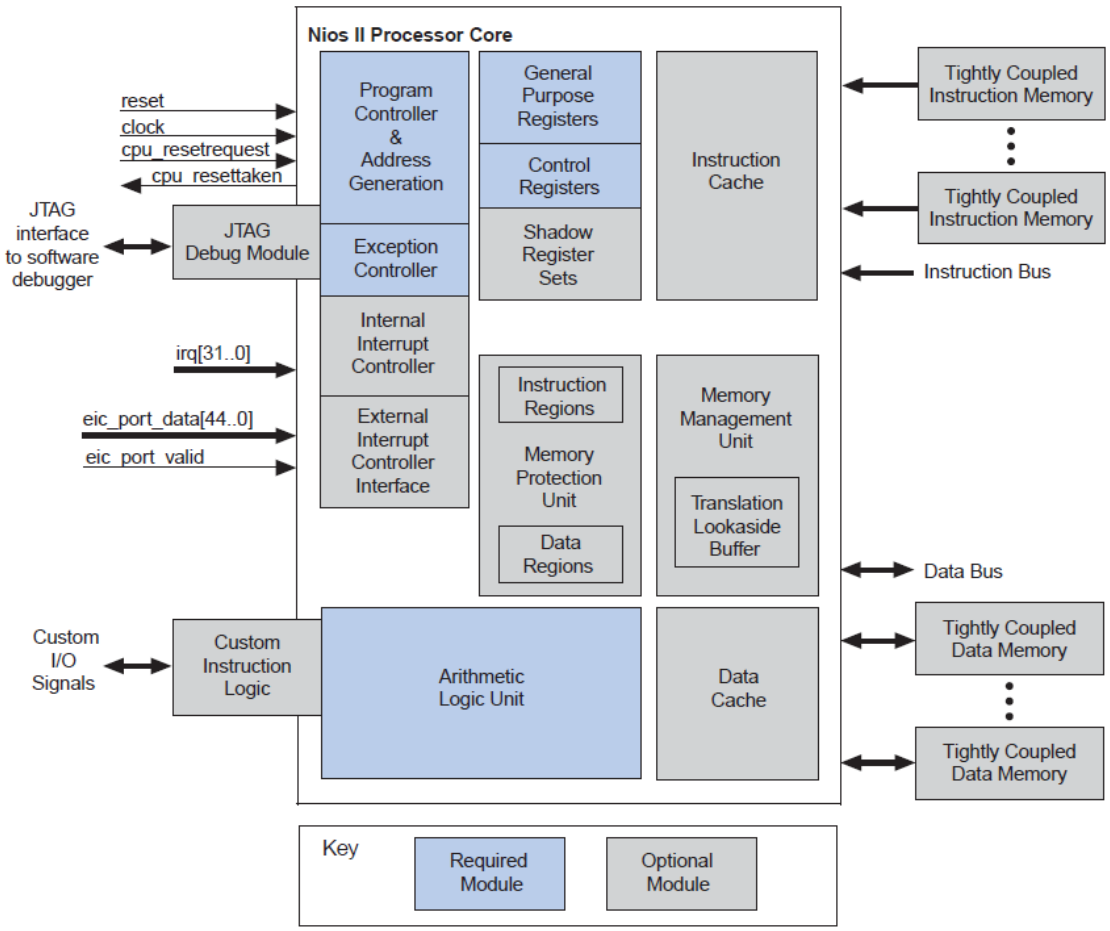
Архитектура Nios II представлена структурой системных команд (ISA). ISA окружена необходимым набором функциональных модулей, исполняющих инструкции. Ядро процессора Nios II – это аппаратный проект, реализующий набор инструкций Nios II и поддерживающий функциональные модули, описанные в этом документе. Ядро процессора не имеет периферии или логики подключения к внешнему миру. Оно содержит только схемы реализации архитектуры Nios II.

На рисунке 2-1 показана блок-схема ядра процессора Nios II.

Архитектура Nios II определяется следующими функциональными модулями:
■ Регистровый файл
■ Арифметико-логическое устройство (ALU)
■ Интерфейс с логикой пользовательских инструкций
■ Контроллер исключений
■ Внутренний или внешний контроллер прерываний
■ Шина данных
■ Диспетчер памяти (MMU)
■ Элемент защиты памяти (MPU)
■ Кэш память под инструкции и данные
■ Сдвоенный интерфейс памяти для инструкций и данных
■ Отладочный модуль JTAG

В следующих секциях обсуждаются детали аппаратной реализации каждого функционального модуля.

Figure 2–1. Nios II Processor Core Block Diagram



Реализация процессора

Функциональные модули архитектуры Nios II формируют основной набор инструкций Nios II. Однако, они не отображаются при аппаратной реализации. Архитектура Nios II описывается набором инструкций, без какой-либо аппаратной реализации. Функциональные модули могут быть аппаратно реализованы, эмулированы программно или полностью пропущены.

Реализация Nios II – это выбор из набора проектов объединённых в специфическом ядре Nios II. Все реализации поддерживают набор инструкций, определённых в главе *Справка по набору инструкций* в *Настольной книге процессора Nios II*. Каждая реализация получает на выходе специфические объекты, например, ядра маленькие по размеру или высокопроизводительные ядра. Это позволяет адаптировать архитектуру Nios II в соответствии с потребностями приложений различных назначений.

В окончательной реализации выбирается один из трёх компромиссных вариантов: больше или меньше средств, включение или исключение средств, аппаратная реализация или программная эмуляция средств.

Проиллюстрируем примерами каждый вариант:

■ Больше или меньше средств – например, для точного выполнения, вы можете увеличивать или уменьшать количество инструкций в кэш памяти. Большой кэш увеличивает скорость исполнения больших программ, тогда как малый кэш сберегает ресурсы внутри чиповой памяти.

■ Включение или исключение средств – например, для уменьшения себестоимости, вы можете не включать отладочный модуль JTAG. Этим вы сбережёте внутри чиповую логику и память, но вам необходимо будет использовать программный отладчик для приложений отладки.

■ Аппаратная реализация или программная эмуляция средств – например, для контрольных приложений, которые редко выполняют сложную арифметику, вы можете выбрать программную эмуляцию инструкций деления. Удаление аппаратного деления сберегает внутри чиповые ресурсы, но увеличивает время выполнения операции деления.

Подробнее о том, какие ядра Nios II поддерживают какие средства, содержится в *Подробностях реализации ядра Nios II* – главе в *Настольной книге процессора Nios II*. Полная информация о выбираемых пользователем параметрах для процессора Nios II, содержится в *Инсталляция процессора Nios II в SOPC Builder*– главе в *Настольной книге процессора Nios II*.

Регистровый файл

Архитектура Nios II поддерживает однородный регистровый файл, состоящий из сдвоенных 32-битных целочисленных регистров общего назначения и дополнительных сдвоенных 32-битных контрольных регистров. Архитектура поддерживает привилегированный и пользовательский режимы, которые позволяют системному коду защищать контрольные регистры от сбившихся приложений.

Процессор Nios II может опционально иметь один или несколько наборов теневых регистров. Набор теневых регистров – это полный набор регистров общего назначения Nios II. Когда реализуется набор теневых регистров, поле CRS статусного регистра показывает текущий набор используемых регистров. Доступ инструкций к регистру общего назначения использует любой набор активных регистров.

Типовое использование набора теневых регистров – ускорение контекстного переключения. Когда реализован набор теневых регистров, процессор Nios II имеет две специальные инструкции, *rdprs* и *wrprs*, для перемещения данных между наборами регистров. Набор теневых регистров обычно управляется ядром операционной системы, поэтому не заметен в коде приложений. Процессор Nios II может иметь до 63 наборов теневых регистров.

Подробнее о реализации и использовании набора теневых регистров, содержится в *"Регистры" и "Процесс исполнения" в Программной модели* – главе в *Настольной книге процессора Nios II*. Полная информация об инструкциях *rdprs* и *wrprs*, содержится в *Справке о наборе инструкций* – главе в *Настольной книге процессора Nios II*.

Архитектура Nios II позволяет дальнейшее добавление регистров с плавающей точкой.

Арифметико-логическое устройство

Nios II ALU оперирует с данными, лежащими в регистрах общего назначения. ALU оперирует с одним или двумя входами регистра и возвращает результат в регистр. ALU поддерживает операции над данными, показанные в таблице 2-1.

Таблица 2-1. Операторы, поддерживаемые Nios II ALU

Категория	Подробности
Арифметические	ALU поддерживает: сложение, вычитание, умножение и деление знаковых и беззнаковых операндов.
Отношения	ALU поддерживает: равенство, неравенство, больше или равно и меньше или равно (==, != >=, <) знаковых и беззнаковых операндов.
Логические	ALU поддерживает: AND, OR, NOR и XOR логические операнды.
Сдвиговые и циклические	ALU поддерживает сдвиговые и циклические операции, и может (циклически) сдвигать данные от 0 до 31 бита позиции в инструкции. ALU поддерживает арифметический сдвиг вправо и логический сдвиг вправо/влево. ALU поддерживает циклический сдвиг вправо/влево.

Для реализации некоторых других операций, программа вычисляет результат выполнения комбинации основных операций из таблицы 2-1.

Одиночные инструкции

Некоторые реализации ядра процессора Nios II не имеют аппаратной поддержки определённого набора инструкций. Таким образом, инструкции ядра без аппаратной поддержки известны как одиночные инструкции.

Процессор генерирует исключение, когда попадает на одиночную инструкцию, так что ваш обработчик исключений может вызвать процедуру, эмулирующую операцию в программе. Поэтому одиночные инструкции не влияют на представление программиста о процессоре.

За списком потенциальных одиночных инструкций обратитесь к *Программная модель* – главе в *Настольной книге процессора Nios II*.

Собственные инструкции

Архитектура Nios II поддерживает определённые пользователем собственные инструкции. Nios II ALU подключена напрямую к логике собственных инструкций, позволяя вам реализовывать аппаратно операции, которые обращаются и используют в точности похожие собственные инструкции.

За подробной информацией обратитесь к *Руководству пользователя по собственным инструкциям Nios II.*

Инструкции с плавающей точкой

Архитектура Nios II поддерживает инструкции с плавающей точкой одинарной точности, как это определено в IEEE Std 754-1985. В базовый набор собственных инструкций с плавающей точкой входят: сложение, вычитание и умножение. Эти инструкции с плавающей точкой рассматриваются как собственные инструкции. В таблице 2-2 приводится подробное описание соответствия с IEEE 754-1985.

Таблица 2-2. Аппаратная совместимость с IEEE 754-1985 стандартом для плавающей точки.

Средство		Реализация
Операции (1)	сложение	реализовано
	вычитание	реализовано
	умножение	реализовано
	деление	опционально
Точность	одинарная	реализовано
	удвоенная	Не реализовано. Удвоенная точность операций реализуется программно.
Исключительная ситуация	неверная операция	Результат – Нет номера (NaN)
	деление на нуль	Результат - ± бесконечность
	переполнение	Результат - ± бесконечность
	не точность	Результат – нормальный номер
	исчезновение данных	Результат - ±0
Режимы округления	округление до ближайшего	реализовано
	округление до нуля	не реализовано
	округление до +∞	не реализовано
	округление до -∞	не реализовано
NaN	исправимый	реализовано
	сигнальный	не реализовано
Субнормальные (денормализованные) числа		Субнормальные операторы рассматриваются как нулевые. Собственные инструкции не генерируют субнормальные числа
Программное исключение		Не реализовано. IEEE 754-1985 состояния исключения детектируются и обрабатываются как показано в этой таблице.
Флаги статуса		Не реализовано. IEEE 754-1985 состояния исключения детектируются и обрабатываются как показано в этой таблице.

Примечание к таблице 2-2:

(1) EDS Nios II позволяет программную реализацию простых операций с плавающей точкой, не считая сложение, вычитание, умножение и деление. Сюда относятся операции преобразования и сравнения. Программная реализация этих примитивов на 100% соответствует IEEE 754-1985.

Вы можете добавить собственные инструкции с плавающей точкой для любого ядра процессора Nios II через интерфейс MegaWizard™. Делению с плавающей точкой требуется больше аппаратных ресурсов, чем другим инструкциям. Интерфейс MegaWizard пропустить аппаратное деление с плавающей точкой для случаев, когда запуск кода в вашем аппаратном проекте не слишком усложняет использование деления с плавающей точкой. Когда вы пропускаете инструкции деления с плавающей точкой, компилятор Nios II реализует программное деление с плавающей точкой.

Для добавления собственных инструкций для вашего процессорного ядра Nios II, обратитесь к *"Странице собственных инструкций"* в главе *Инсталляция процессора Nios II в SOPC Builder* – главе в *Настольной книге процессора Nios II*.

Собственные инструкции с плавающей точкой для процессора Nios II основаны на мегафункциях с плавающей точкой Altera®.

Подробнее о каждой конкретной мегафункции с плавающей точкой, включая факторы ускорения и использование ресурсов чипа, обратитесь к руководствам пользователя конкретных мегафункций, доступным на странице *Литература по IP и мегафункциям* на веб-сайте Altera.

Инструмент разработки программы под Nios II распознаёт C код, который выгодно использует описание инструкций с плавающей точкой, в процессорном ядре. Когда собственные инструкции с плавающей точкой представлены в вашей аппаратной части, компилятор Nios II компилирует ваш код для использования собственных инструкций для операций с плавающей точкой, включая сложение, вычитание, умножение, деление и математическую библиотеку newlib.

Рассмотрение разработки программы

Наилучшим выбором для вашего аппаратного проекта будет баланс между использованием плавающей точки, использованием аппаратных ресурсов и рабочими характеристиками. Когда собственные инструкции с плавающей точкой ускоряют арифметику с плавающей точкой, это подразумевает увеличение размеров вашего аппаратного проекта. Если использование ресурсов ограничено, приходится перерабатывать ваши алгоритмы для минимизации арифметики с плавающей точкой.

Вы можете использовать директивы **#pragma** в вашей программе для сравнения аппаратной и программной реализаций инструкций с плавающей точкой. Следование директивам **#pragma** инструктирует компилятор Nios II игнорировать инструкции с плавающей точкой и генерировать их программную реализацию. Область действия директивы **#pragma** – целый C файл.

- #pragma no_custom_fadds— Принуждает программу реализовать сложение.
- #pragma no_custom_fsubs— Принуждает программу реализовать вычитание.
- #pragma no_custom_fmuls— Принуждает программу реализовать умножение.
- #pragma no_custom_fdivs— Принуждает программу реализовать деление.

Система команд симулятора (ISS) Nios II не поддерживает собственные инструкции. Если вам необходимо запустить вашу программу в ISS, запретите собственные инструкции с плавающей точкой с помощью директивы **#pragma**.

Все собственные инструкции с плавающей точкой являются операторами с одинарной точностью. Операторы с двойной точностью реализуются программно. По умолчанию, компилятор Nios II рассматривает константы с плавающей точкой как числа с двойной точностью. Для использования собственных инструкций с плавающей точкой в операторах с константами с плавающей точкой, добавьте "**f**" константе. Это обяжет компилятор рассматривать константу как число с одинарной точностью, а остальным переменным в ваших вычислениях останется двойная точность чисел. В таблице 2-3 показаны примеры кода, использующего константы.

Таблица 2-3. Примеры констант с плавающей точкой

Пример кода	**Точность**	**Использование собственных инструкций с плавающей точкой**
y = x × 4.67;	двойная	нет
y = x × 4.67f;	одинарная	да