Дипломная работа

Разработка RTL-описания интегрированного микропроцессорного модуля с RISC-архитектурой

Кареев Кирилл Андреевич[1]

[1]МИЭТ

МИЭТ, 2016

Содержание

1. Мотвация
   - Почему Я Решил Сделать Ещё Один RISC Процессор
   - На Что Я Опирался
2. Процессор УП-1
   - Общее описание
   - Набор инструкций

1 Мотвация
- Почему Я Решил Сделать Ещё Один RISC Процессор
- На Что Я Опирался

2 Процессор УП-1
- Общее описание
- Набор инструкций

Чего не хватает другим процессорам

- Простоты работы с машинным кодом и ассемблерным представлением.
- Единой структуры блока исполнения, без сопроцессоров и особых регистровых наборов.
- Открытости RTL-описания.

Чего не хватает другим процессорам

- Простоты работы с машинным кодом и ассемблерным представлением.
- Единой структуры блока исполнения, без сопроцессоров и особых регистровых наборов.
- Открытости RTL-описания.

Чего не хватает другим процессорам

- Простоты работы с машинным кодом и ассемблерным представлением.
- Единой структуры блока исполнения, без сопроцессоров и особых регистровых наборов.
- Открытости RTL-описания.

Что есть в моём процессоре

- Развитый набор команд, позволяющий в большинстве случаев обойтись без макросов и теневых определений.
- Практически все инструкции исполняются за один такт.
- Все инструкции могут использовать все регистры и иметь мгновенные значения для всех входных аргументов.
- Все инструкции относятся к основному набору, т.е. в процессоре нет сопроцессоров.
- Составные блоки конвейера достаточно универсальны для того, чтобы новый набор инструкций можно было ввести сменой лишь одного декодера.
- Открытое RTL-описание - читать и править можно всем :)

Что есть в моём процессоре

- Развитый набор команд, позволяющий в большинстве случаев обойтись без макросов и теневых определений.
- Практически все инструкции исполняются за один такт.
- Все инструкции могут использовать все регистры и иметь мгновенные значения для всех входных аргументов.
- Все инструкции относятся к основному набору, т.е. в процессоре нет сопроцессоров.
- Составные блоки конвейера достаточно универсальны для того, чтобы новый набор инструкций можно было ввести сменой лишь одного декодера.
- Открытое RTL-описание - читать и править можно всем :)

Что есть в моём процессоре

- Развитый набор команд, позволяющий в большинстве случаев обойтись без макросов и теневых определений.
- Практически все инструкции исполняются за один такт.
- Все инструкции могут использовать все регистры и иметь мгновенные значения для всех входных аргументов.
- Все инструкции относятся к основному набору, т.е. в процессоре нет сопроцессоров.
- Составные блоки конвейера достаточно универсальны для того, чтобы новый набор инструкций можно было ввести сменой лишь одного декодера.
- Открытое RTL-описание - читать и править можно всем :)

Что есть в моём процессоре

- Развитый набор команд, позволяющий в большинстве случаев обойтись без макросов и теневых определений.
- Практически все инструкции исполняются за один такт.
- Все инструкции могут использовать все регистры и иметь мгновенные значения для всех входных аргументов.
- Все инструкции относятся к основному набору, т.е. в процессоре нет сопроцессоров.
- Составные блоки конвейера достаточно универсальны для того, чтобы новый набор инструкций можно было ввести сменой лишь одного декодера.
- Открытое RTL-описание - читать и править можно всем :)

Что есть в моём процессоре

- Развитый набор команд, позволяющий в большинстве случаев обойтись без макросов и теневых определений.
- Практически все инструкции исполняются за один такт.
- Все инструкции могут использовать все регистры и иметь мгновенные значения для всех входных аргументов.
- Все инструкции относятся к основному набору, т.е. в процессоре нет сопроцессоров.
- Составные блоки конвейера достаточно универсальны для того, чтобы новый набор инструкций можно было ввести сменой лишь одного декодера.
- Открытое RTL-описание - читать и править можно всем :)

Чего нет в моём процессоре

- Достаточного функционала для реализации какого-либо CISC-набора инструкций.
- Векторных расширений и работы с плавающими числами двойной и четверной точности.
- Упора на малый размер бинарного представления машинного кода.
- Мультиплексирования памяти по частям машинного слова (возможна работа только с целым словом).
- Особых свойств (флаги, префиксы) команд и состояний (ошибка, гипервизор, быстрое прерывание и т.д.) процессора

Чего нет в моём процессоре

- Достаточного функционала для реализации какого-либо CISC-набора инструкций.
- Векторных расширений и работы с плавающими числами двойной и четверной точности.
- Упора на малый размер бинарного представления машинного кода.
- Мультиплексирования памяти по частям машинного слова (возможна работа только с целым словом).
- Особых свойств (флаги, префиксы) команд и состояний (ошибка, гипервизор, быстрое прерывание и т.д.) процессора

Чего нет в моём процессоре

- Достаточного функционала для реализации какого-либо CISC-набора инструкций.
- Векторных расширений и работы с плавающими числами двойной и четверной точности.
- Упора на малый размер бинарного представления машинного кода.
- Мультиплексирования памяти по частям машинного слова (возможна работа только с целым словом).
- Особых свойств (флаги, префиксы) команд и состояний (ошибка, гипервизор, быстрое прерывание и т.д.) процессора

Чего нет в моём процессоре

- Достаточного функционала для реализации какого-либо CISC-набора инструкций.
- Векторных расширений и работы с плавающими числами двойной и четверной точности.
- Упора на малый размер бинарного представления машинного кода.
- Мультиплексирования памяти по частям машинного слова (возможна работа только с целым словом).
- Особых свойств (флаги, префиксы) команд и состояний (ошибка, гипервизор, быстрое прерывание и т.д.) процессора

Чего нет в моём процессоре

- Достаточного функционала для реализации какого-либо CISC-набора инструкций.
- Векторных расширений и работы с плавающими числами двойной и четверной точности.
- Упора на малый размер бинарного представления машинного кода.
- Мультиплексирования памяти по частям машинного слова (возможна работа только с целым словом).
- Особых свойств (флаги, префиксы) команд и состояний (ошибка, гипервизор, быстрое прерывание и т.д.) процессора

1 Мотвация
- Почему Я Решил Сделать Ещё Один RISC Процессор
- На Что Я Опирался

2 Процессор УП-1
- Общее описание
- Набор инструкций

Существующие решения
ARM Thumb1

+ Простота (функциональная) набора инструкций.
+ Простота строения процессора.
+ Большая база поддержки (компиляторы, ОС etc.).
- Сложность бинарного представления машинного кода (из-за упора на уменьшенный размер).
- Работа с дробными частями машинного слова.
- Сложность работы с ассемблерным представлением кода (следствие функциональной простоты).
- Расширения функционала (работа с плавающей точкой, например) перенесены на сопроцессоры и ARM режим.

Существующие решения
ARM Thumb1

+ Простота (функциональная) набора инструкций.
+ Простота строения процессора.
+ Большая база поддержки (компиляторы, ОС etc.).
- Сложность бинарного представления машинного кода (из-за упора на уменьшенный размер).
- Работа с дробными частями машинного слова.
- Сложность работы с ассемблерным представлением кода (следствие функциональной простоты).
- Расширения функционала (работа с плавающей точкой, например) перенесены на сопроцессоры и ARM режим.

Существующие решения
OpenRISC 1000 (mor1kx)

+ Полная открытость как RTL-описания, так и базы поддержки.
+ Относительная простота набора инструкций.
+ Наличие расширений для работы с плавающей точкой.
+ Большая база поддержки.
- Наличие большого количества состояний процессора.
- Расширения так или иначе вынесены в сопроцессор.
- Сложность RTL-описания, в основном из-за высокой функциональной развитости.

Существующие решения
OpenRISC 1000 (mor1kx)

+ Полная открытость как RTL-описания, так и базы поддержки.
+ Относительная простота набора инструкций.
+ Наличие расширений для работы с плавающей точкой.
+ Большая база поддержки.
- Наличие большого количества состояний процессора.
- Расширения так или иначе вынесены в сопроцессор.
- Сложность RTL-описания, в основном из-за высокой функциональной развитости.

Существующие решения
MIPS32

+ Существует много реализаций, в том числе и свободных
+ Большая база поддержки, проверен временем
+ Малый размер набора инструкций
+ Наличие большого количества расширений, в тч и для работы с плавающей точкой
- Отночительно сложное построение инструкции
- Большинство расширений помещено в сопроцессоры
- Далеко не все реализации соответствуют стандарту

Существующие решения
MIPS32

+ Существует много реализаций, в том числе и свободных
+ Большая база поддержки, проверен временем
+ Малый размер набора инструкций
+ Наличие большого количества расширений, в тч и для работы с плавающей точкой
- Отночительно сложное построение инструкции
- Большинство расширений помещено в сопроцессоры
- Далеко не все реализации соответствуют стандарту

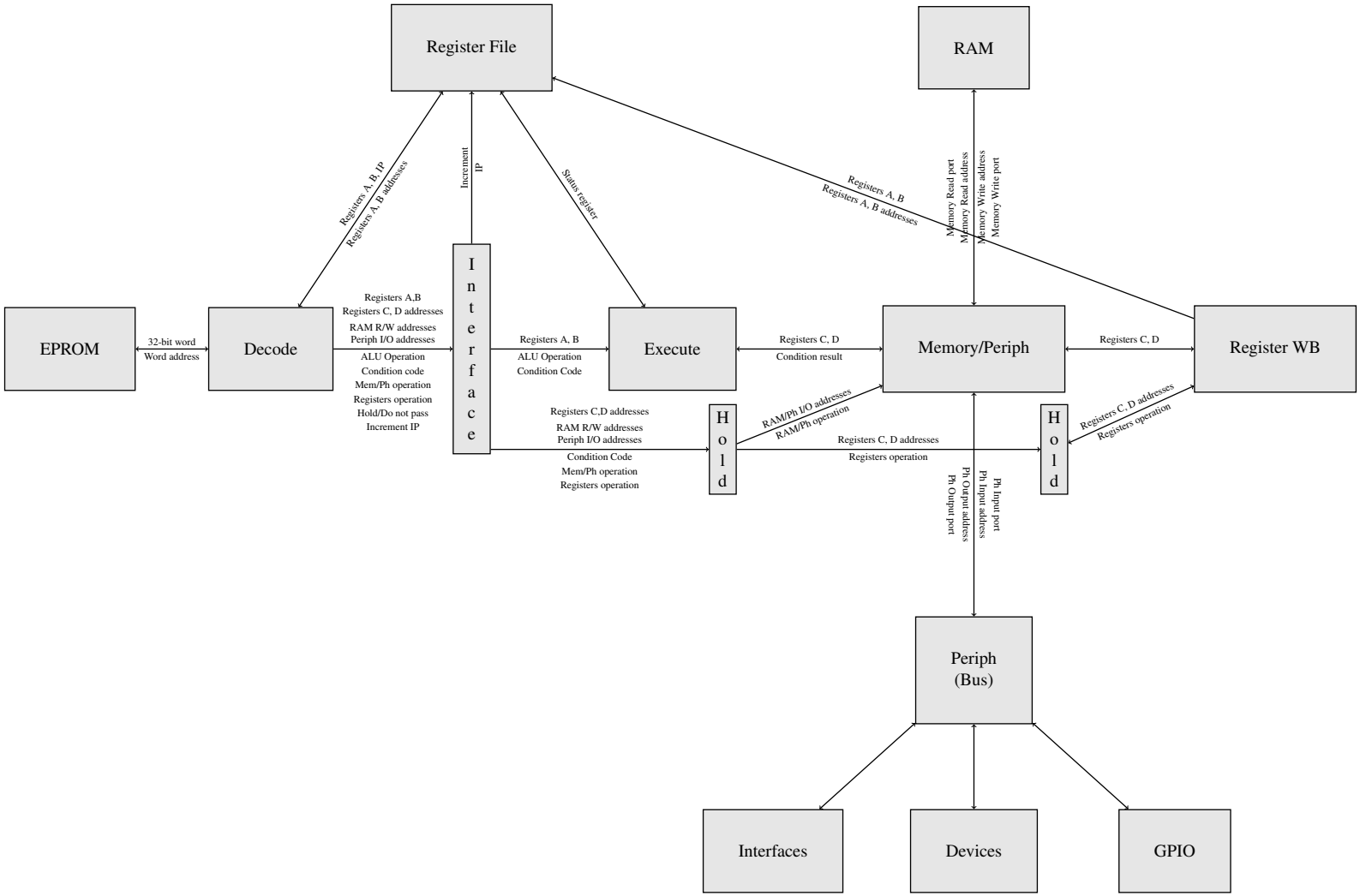
1 Мотвация
- Почему Я Решил Сделать Ещё Один RISC Процессор
- На Что Я Опирался

2 Процессор УП-1
- Общее описание
- Набор инструкций

Характеристики

- 32-битная архитектура
- Набор из 64 (заложено до 128) инструкций
- 32 РОН (Регистра общего назначения) шириной 32 бита с четырёхпортовым интерфейсом (2 чтение, 2 запись + особые линии для PC, LR и SP)
- Регистры PC, LR и SP (счётчик инструкций, адрес возврата и указатель стека) также являются общими (29, 30 и 31 соответственно)
- Однотактовый умножитель с возможностью сохранения всего результата (2 слова)
- Однотактовый комбинированный регистр сдвига (циклический, арифметический и обыкновенный сдвиг)
- Раздельные шины памяти и периферический устройств
- 16 кодов условного исполнения
- Четырёхшаговая архитектура конвейера (Декодирование, Исполнение, Память/Периферия и Регистры)
- 4 кб (с возможностью увеличения) двухпортового однотактового ОЗУ (1- чтение, 1 - запись)
- 16 кб (с возможностью увеличения) однопортового однотактового (как бы) ППЗУ

Структурная схема



Структурная схема
Устройства

- EPROM - ППЗУ. На выходе 32-битное слово, находящееся по адресу на входе. Текущий размер - 16 кб (4096 слов)
- Register File - регистровый файл. Имеет 32 регистра, 2 порта на чтение (A, B) и 2 порта на запись (C, D) и спецвыходы для IP и Status
- RAM - ОЗУ. Имеет 1 порт на запись и 1 порт на чтение. Текущий размер - 4 кб (1024 слов).
- Periph - Двунаправленная шина периферических устройств. Имеет два раздельных (адрес+данные) направления - запись и чтение. К ней подключены:
  - GPIO - Ввод-вывод общего назначения - «ноги», с мультиплексорами для выбора дополнительных функций.
  - Interfaces - Различные интерфейсы ввода-вывода (SPI и UART, например)
  - Devices - Различные периферические устройства (RTC и ШИМ-контроллеры, например)

Структурная схема
Стадии конвейера

- Decode - стадия дешифрования входного слова (инструкции) в набор управляющих сигналов. (D)
  - Interface - специальная стадия задержки, нужна для осуществления многотактовых инструкций. (I)
- Execute - стадия исполнения инструкции. Содержит АЛУ и блок вычисления условных кодов. (EX)
- Memory/Periph - стадия комбинированных операций с памятью и шиной периферических устройств. (MEM)
- Register WB - стадия обратной записи в регистры.
- HOLD(n) - специальные подстадии, обеспечивающие удержание управляющих сигналов для синхронизации.

Схема АЛУ

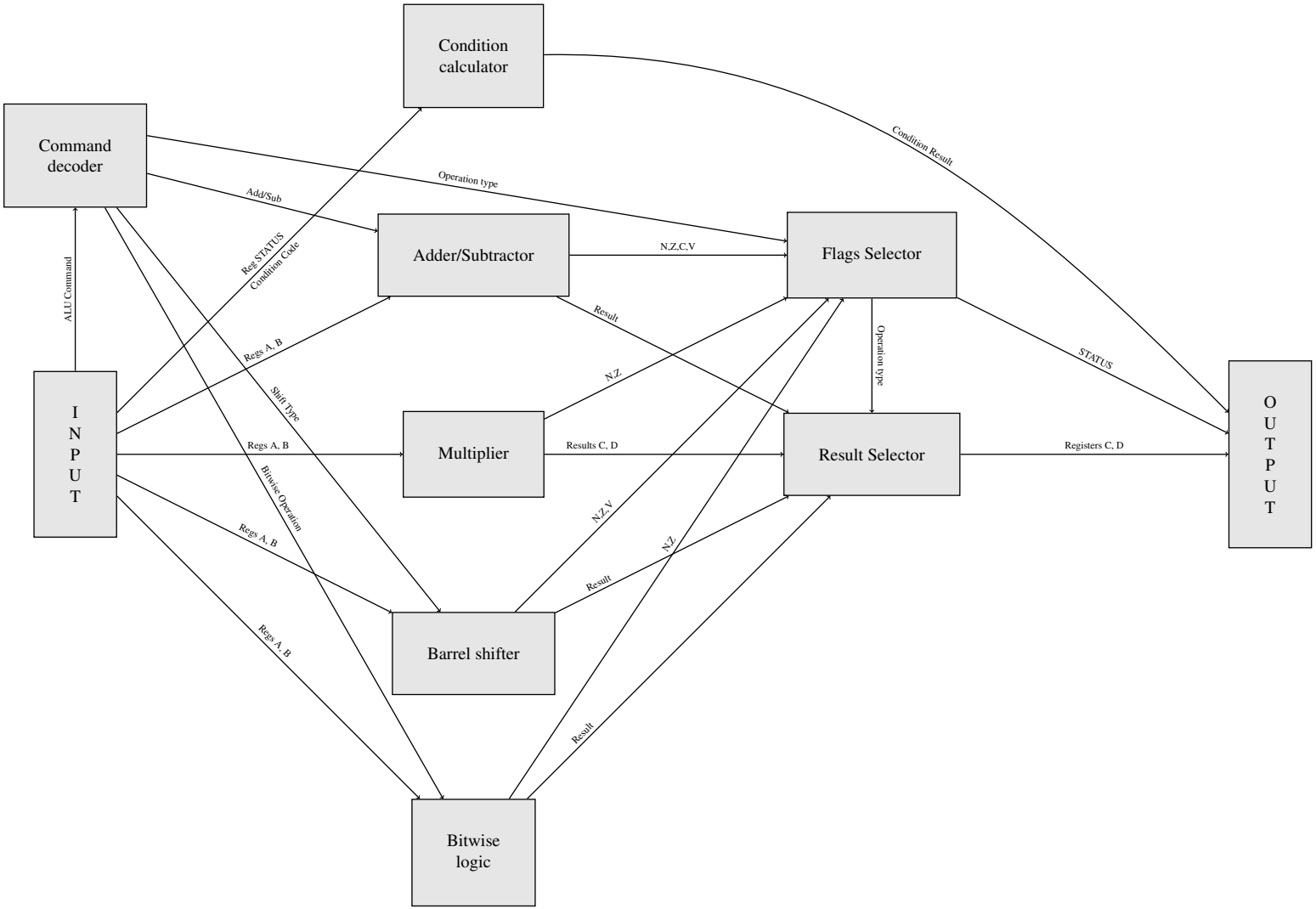

Condition calculator
Command decoder
Condition Result
Operation type
Add/Sub
Adder/Subtractor
N,Z,C,V
Flags Selector
Reg STATUS Condition Code
ALU Command
Result
Regs A, B
N,Z,
Operation type
STATUS
I N P U T
Shift Type
Multiplier
Result Selector
O U T P U T
Regs A, B
Results C, D
Registers C, D
Bitwise Operation
N,Z,V
Regs A, B
N,Z,
Result
Barrel shifter
Regs A, B
Result
Bitwise logic

Схема АЛУ
Устройства

- Command Decoder - декодер команды АЛУ во внутренние сигналы управления
- Condition calculator - вычисляет условный результат из регистра STATUS и условного кода
- Adder/Subtractor - комбинированный сумматор-вычитатель 32x32=33
- Multiplier - беззнаковый умножитель 32x32=64
- Barrel shifter - комбинированный сдвиговый регистр 32x5=33
- Bitwise logic - комбинированный блок логических операций, 32x32=32

Флаги (статус)

- (N)egative - отрицательное число, равен 31 биту результата
- (Z)ero - нулевой результат
- (C)arry - беззнаковое переполнение (перенос), например, FFFFFFFFh + FFFFFFFFh (беззнаковое) = C + FFFFFFFE
- O(V)erflow - знаковое переполнение, например, 7FFFFFFF + 7FFFFFFF (знаковое) = V - 7FFFFFFE

1 Мотвация
- Почему Я Решил Сделать Ещё Один RISC Процессор
- На Что Я Опирался

2 Процессор УП-1
- Общее описание
- Набор инструкций

Характеристики

- 6 логических инструкций (or, nor, and, nand, inv, xor).
- 6 видов сдвига (арифметический, логический и циклический), влево и вправо.
- Арифметический операции (сумма, разность, беззнаковое произведение, инкремент/декремент, сравнение)
- Операции потока исполнения (прыжок, вызов и возврат), в т.ч и относительные
- Операции с памятью и стеком (чтение и запись), в т.ч. непрямые
- Операция перемещения регистр-регистр (в т.ч двойная) и пустая операция
- Операции с шиной периферических устройств
- Все операции имеют возможность условного исполнения.

Условные коды

- EQ - равен (первый операнд второму)
- NEQ - не равен
- HE - больше либо равен
- H - больше
- LE - меньше либо равен
- L - меньше
- SHE - знаковый больше либо равен
- SH - знаковый больше
- SLE - знаковый меньше либо равен
- SL - знаковый меньше
- OV - переполнение (знаковое)
- NOV - не переполнение (знаковое)
- POS - положительное число
- NEG - отрицательное число
- AL - всегда (безусловно)

Перечисление инструкций

Логические инструкции

- OR - побитовое ИЛИ двух операндов в третий
- NOR - побитовое ИЛИ-НЕ двух операндов в третий
- AND - побитовое И двух операндов в третий
- NAND - побитовое И-НЕ двух операндов в третий
- INV - побитовая инверсия одного операнда во второй
- XOR - побитовое ИСКЛ. ИЛИ двух операндов в третий
- XNOR - побитовое ИСКЛ. ИЛИ-НЕ двух операндов в третий

Перечисление инструкций

Сдвиговые инструкции

- LSL - логический сдвиг влево
- LSR - логический сдвиг вправо
- ASR - арифметический сдвиг вправо
- ASL - арифметический сдвиг влево
- CSR - циклический сдвиг (вращение) вправо
- CSL - циклический сдвиг (вращение) влево

Перечисление инструкций
Арифметические операции

- ADD - сложение двух операндов в третий
- SUB - вычитание двух операндов в третий
- MUL - беззнаковое перемножение двух операндов в третий(LO) и четвёртый (HI)
- CSG - смена знака операнда во второй
- INC - инкремент операнда во второй
- DEC - декремент операнда во второй
- CMP - сравнение двух операндов (вычитание без сохранения результата)
- CMN - сравнение операнда с инвертированнным вторым.
- TST - тест операндов (побитовое И без сохранения результата)

Перечисление инструкций

Переходы

- BR - абсолютный переход по адресу в операнде
- RBR - относительный (малый) переход по адресу в IP+операнд
- BRL - абсолютный переход с запоминанием (в LR)
- RBL - относительный переход с запоминанием (в LR)
- RET - возврат (переход по адресу в LR)

Перечисление инструкций

Память и стек

- LDR - загрузка из адреса в операнде
- STR - запись по адресу в операнде
- LDRC - непрямая загрузка из адреса операнд1+операнд2
- STRC - непрямая запись по адресу операнд1+операнд2
- PUSH - помещение операнда на стек
- POP - извлечение слова со стека и помещение в операнд

Перечисление инструкций

Периферия и прочее

- IN - ввод из регистра периферии по адресу в операнде
- OUT - вывод в регистр периферии по адресу в операнде
- INI - непрямой ввод из регистра периферии по адресу операнд1+операнд2
- OUTI - непрямой вывод в регистр периферии по адресу операнд1+операнд2
- MOV - скопировать {операнд1, операнд2} в {операнд3, операнд 4}
- NOP - ничего не делать 1 (или операнд1) такт(ов).

Результат

- Создано RTL-описание процессора УП-1.
- Создана документация к процессору УП-1 .
- Произведено моделирование процессора УП-1 в симуляторе Icarus Verilog.

- Что дальше
  - Добавить инструкции работы с плавающей точкой одинарной точности.
  - Добавить какую-нибудь периферию кроме GPIO и сдвигового регистра («SPI»).
  - Добавить кэш и спекулятивное исполнение.

Результат

- Создано RTL-описание процессора УП-1.
- Создана документация к процессору УП-1 .
- Произведено моделирование процессора УП-1 в симуляторе Icarus Verilog.

- Что дальше
  - Добавить инструкции работы с плавающей точкой одинарной точности.
  - Добавить какую-нибудь периферию кроме GPIO и сдвигового регистра («SPI»).
  - Добавить кэш и спекулятивное исполнение.