Связанный с инструкцией обработчик исключений

Направитель (funnel) программных исключений позволяет вам обрабатывать связанные с инструкцией исключения, такие как расширенные исключения. Обработчик связанных исключений является собственным обработчиком. Ваша программа регистрирует связанный с инструкцией обработчик исключений с помощью HAL во время запуска.

Настройка hal.enable_instruction_related_exceptions_api должна разрешена в BSP, чтобы вы смогли зарегистрировать связанный с инструкцией обработчик исключений.

За дополнительной информацией о связанном с инструкцией обработчике исключений обратитесь к главе "Программная модель" в настольной книге по процессору Nios II. За подробной информацией о разрешении связанного с инструкцией обработчика исключений, обратитесь к секции "Настройки" в главе "Справка по инструменту создания программы под Nios II" в настольной книге программиста Nios II.

Когда вы зарегистрируете связанный с инструкцией обработчик исключений, он зарезервирует место под логику останова / системного прерывания.

Когда вы удаляете связанный с инструкцией обработчик исключений, HAL восстанавливает логику останова / системного прерывания по умолчанию.

Написание связанного с инструкцией обработчика исключений

Прототип для связанного с инструкцией обработчика исключений следующий:

```
alt_exception_result handler (
  alt_exception_cause  cause,
  alt_u32              addr,
  alt_u32               bad_addr );
```

Связанный с инструкцией обработчик исключений возвращает значение – флаг запроса, который HAL использует для повторного исполнения инструкции или пропускает его.

Направитель (funnel) исключения вызывает связанный с инструкцией обработчик исключений со следующими аргументами:

- cause – значение, представляющее собой тип исключения, как показано в табл. 8-4.
- addr – адрес инструкции, по которому происходит исключение.
- bad_addr – регистр плохого адреса (если он реализован)

Включите следующий заголовочный файл в ваш код связанного с инструкцией обработчика исключений:

#include “sys/alt_exceptions.h”

Файл **alt_exceptions.h** предлагает тип макро определений, требуемый для интерфейса вашего связанного с инструкцией обработчика исключений с HAL, включая коды вызова, показанные в табл. 8-4.

API функция alt_exception_cause_generated_bad_addr() предлагается HAL для использования в связанном с инструкцией обработчике исключений. Эта функция анализирует аргумент вызова и определяет, когда bad_addr содержит адрес вызова исключений.

За дополнительной информацией о вызове исключений в процессоре Nios II, обратитесь к секции "Процесс исключений" в главе "Программная модель" в настольной книге по процессору Nios II.

Табл. 8-4 Коды вызова исключений процессора Nios II

Исключение	**Код вызова**	**Символ вызова** (1)
Сброс	0	NIOS2_EXCEPTION_RESET
Запрос сброса только процессора	1	NIOS2_EXCEPTION_CPU_ONLY_RESET_REQUEST
Аппаратное прерывание	2	NIOS2_EXCEPTION_INTERRUPT
Инструкция системного прерывания	3	NIOS2_EXCEPTION_TRAP_INST
Нереализуемая инструкция	4	NIOS2_EXCEPTION_UNIMPLEMENTED_INST
Неверная инструкция	5	NIOS2_EXCEPTION_ILLEGAL_INST
Смещённый адрес данных	6	NIOS2_EXCEPTION_MISALIGNED_DATA_ADDR
Смещённый адрес назначения	7	NIOS2_EXCEPTION_MISALIGNED_TARGET_PC
Ошибка деления	8	NIOS2_EXCEPTION_DIVISION_ERROR
Адрес инструкции управляющей программы	9	NIOS2_EXCEPTION_SUPERVISOR_ONLY_INST_ADDR
Инструкция управляющей программы	10	NIOS2_EXCEPTION_SUPERVISOR_ONLY_INST
Адрес данных инструкции управляющей программы	11	NIOS2_EXCEPTION_SUPERVISOR_ONLY_DATA_ADDR
Потеря буфера быстрого преобразования адреса (TLB)	12	NIOS2_EXCEPTION_TLB_MISS
Нарушение прав TLB (исполнение)	13	NIOS2_EXCEPTION_TLB_EXECUTE_PERM_VIOLATION
Нарушение прав TLB (чтение)	14	NIOS2_EXCEPTION_TLB_READ_PERM_VIOLATION
Нарушение прав TLB (запись)	15	NIOS2_EXCEPTION_TLB_WRITE_PERM_VIOLATION
Нарушение региона MPU (инструкция)	16	NIOS2_EXCEPTION_MPU_INST_REGION_VIOLATION
Нарушение региона MPU (данные)	17	NIOS2_EXCEPTION_MPU_DATA_REGION_VIOLATION
Неизвестный вызов (2)	-1	NIOS2_EXCEPTION_CAUSE_NOT_PRESENT

Примечания к табл. 8-4:
(1) Символы вызова определены в файле **sys/alt_exceptions.h**.
(2) Это значение пропущено в связанном с инструкцией обработчике прерываний, когда в аргументе cause неизвестен вызов; например, когда регистр cause не реализован в ядре процессора Nios II.

Если имеется связанный с инструкцией обработчик исключений, то он вызывается в конце направителя (funnel) программного исключения (если направитель не распознаёт аппаратное прерывание, нереализуемую инструкцию или системное прерывание (trap)). Он занимает часть инструкции останова (break) или непрерывного цикла (infinite loop). Поэтому для поддержки отладки, выполните инструкцию останова или аппаратного прерывания.

Для связанного с инструкцией обработчика исключений возможен вызов во время исполнения ISR.

Регистрирование связанного с инструкцией обработчика исключений

Функция HAL API alt_instruction_exception_register() регистрирует один связанный с инструкцией обработчик исключений.

Прототип функции следующий:

```
alt_instruction_exception_register (
    alt_exception_result (*handler)
        ( alt_exception_cause, alt_u32, alt_u32 ));
```

Аргумент handler – это указатель на связанный с инструкцией обработчик исключений. Чтобы использовать alt_instruction_exception_register(), добавьте следующий заголовочный файл:

```
#include "sys/alt_exceptions.h"
```

Настройка hal.enable_instruction_related_exceptions_api должна разрешена в BSP, чтобы вы смогли зарегистрировать связанный с инструкцией обработчик исключений.

За подробной информацией о разрешении связанного с инструкцией обработчика исключений, обратитесь к секции "Настройки" в главе "Справка по инструменту создания программы под Nios II" в настольной книге программиста Nios II.

Регистрируйте связанный с инструкцией обработчик исключений насколько можно раньше в теле main(). Это позволит вам обрабатывать аномальные состояния во время запуска. Вы регистрируете обработчик исключений из функции alt_main().

За подробной информацией об alt_main(), обратитесь к секции "Последовательность загрузки и точка входа" в главе "Разработка программ с использованием слоя аппаратной абстракции" в настольной книге программиста Nios II.

Удаление связанного с инструкцией обработчика исключений

Для удаления зарегистрированного связанного с инструкцией обработчика исключений, в вашем Си коде нужно вызвать функцию alt_instruction_exception_register() в виде

```
alt_instruction_exception_register ( null, null );
```

Когда HAL удалит связанный с инструкцией обработчик исключений, он восстановит логику останова / дополнительного системного прерывания по умолчанию.