
SuperH architecture implementation manual

Release

The kernel development community

Nov 02, 2017

1	Memory Management	3
1.1	SH-4	3
1.2	SH-5	4
2	Machine Specific Interfaces	7
2.1	mach-dreamcast	7
2.2	mach-x3proto	7
3	Busses	9
3.1	SuperHyway	9
3.2	Maple	10
	Index	11

Author Paul Mundt

MEMORY MANAGEMENT

SH-4

Store Queue API

void **sq_flush_range**(unsigned long *start*, unsigned int *len*)
Flush (prefetch) a specific SQ range

Parameters

unsigned long start the store queue address to start flushing from

unsigned int len the length to flush

Description

Flushes the store queue cache from **start** to **start + len** in a linear fashion.

unsigned long **sq_remap**(unsigned long *phys*, unsigned int *size*, const char * *name*, pgprot_t *prot*)
Map a physical address through the Store Queues

Parameters

unsigned long phys Physical address of mapping.

unsigned int size Length of mapping.

const char * name User invoking mapping.

pgprot_t prot Protection bits.

Description

Remaps the physical address **phys** through the next available store queue address of **size** length. **name** is logged at boot time as well as through the sysfs interface.

void **sq_unmap**(unsigned long *vaddr*)
Unmap a Store Queue allocation

Parameters

unsigned long vaddr Pre-allocated Store Queue mapping.

Description

Unmaps the store queue allocation **map** that was previously created by [*sq_remap\(\)*](#). Also frees up the pte that was previously inserted into the kernel page table and discards the UTLB translation.

SH-5

TLB Interfaces

int **sh64_tlb_init**(void)

Perform initial setup for the DTLB and ITLB.

Parameters

void no arguments

unsigned long long **sh64_next_free_dtlb_entry**(void)

Find the next available DTLB entry

Parameters

void no arguments

unsigned long long **sh64_get_wired_dtlb_entry**(void)

Allocate a wired (locked-in) entry in the DTLB

Parameters

void no arguments

int **sh64_put_wired_dtlb_entry**(unsigned long long *entry*)

Free a wired (locked-in) entry in the DTLB.

Parameters

unsigned long long entry Address of TLB slot.

Description

Works like a stack, last one to allocate must be first one to free.

void **sh64_setup_tlb_slot**(unsigned long long *config_addr*, unsigned long *eaddr*, unsigned long *asid*, unsigned long *paddr*)

Load up a translation in a wired slot.

Parameters

unsigned long long config_addr Address of TLB slot.

unsigned long eaddr Virtual address.

unsigned long asid Address Space Identifier.

unsigned long paddr Physical address.

Description

Load up a virtual<->physical translation for **eaddr**<->**paddr** in the pre-allocated TLB slot **config_addr** (see **sh64_get_wired_dtlb_entry**).

void **sh64_tearardown_tlb_slot**(unsigned long long *config_addr*)

Teardown a translation.

Parameters

unsigned long long config_addr Address of TLB slot.

Description

Teardown any existing mapping in the TLB slot **config_addr**.

for_each_dtlb_entry(*tlb*)

Iterate over free (non-wired) DTLB entries

Parameters

tlb TLB entry

for_each_itlb_entry(*tlb*)

Iterate over free (non-wired) ITLB entries

Parameters

tlb TLB entry

void **__flush_tlb_slot**(unsigned long long *slot*)

Flushes TLB slot **slot**.

Parameters

unsigned long long slot Address of TLB slot.

MACHINE SPECIFIC INTERFACES

`mach-dreamcast`

void **`aica_rtc_gettimeofday`**(struct timespec * *ts*)
Get the time from the AICA RTC

Parameters

`struct timespec * ts` pointer to resulting timespec

Description

Grabs the current RTC seconds counter and adjusts it to the Unix Epoch.

int **`aica_rtc_settimeofday`**(const time_t *secs*)
Set the AICA RTC to the current time

Parameters

`const time_t secs` contains the time_t to set

Description

Adjusts the given **`tv`** to the AICA Epoch and sets the RTC seconds counter.

`mach-x3proto`

int **`ilsel_enable`**(ilsel_source_t *set*)
Enable an ILSEL set.

Parameters

`ilsel_source_t set` ILSEL source (see `ilsel_source_t` enum in `include/asm-sh/ilsel.h`).

Description

Enables a given non-aliased ILSEL source (\leq ILSEL_KEY) at the highest available interrupt level. Callers should take care to order callsites noting descending interrupt levels. Aliasing FPGA and external board IRQs need to use *`ilsel_enable_fixed()`*.

The return value is an IRQ number that can later be taken down with *`ilsel_disable()`*.

int **`ilsel_enable_fixed`**(ilsel_source_t *set*, unsigned int *level*)
Enable an ILSEL set at a fixed interrupt level

Parameters

`ilsel_source_t set` ILSEL source (see `ilsel_source_t` enum in `include/asm-sh/ilsel.h`).

`unsigned int level` Interrupt level (1 - 15)

Description

Enables a given ILSEL source at a fixed interrupt level. Necessary both for level reservation as well as for aliased sources that only exist on special ILSEL#s.

Returns an IRQ number (as *ilssel_enable()*).

void **ilssel_disable**(unsigned int *irq*)
Disable an ILSEL set

Parameters

unsigned int irq Bit position for ILSEL set value (retval from enable routines)

Description

Disable a previously enabled ILSEL set.

SuperHyway

int **superhyway_add_device**(unsigned long *base*, struct superhyway_device * *sdev*, struct superhyway_bus * *bus*)
Add a SuperHyway module

Parameters

unsigned long base Physical address where module is mapped.

struct superhyway_device * sdev SuperHyway device to add, or NULL to allocate a new one.

struct superhyway_bus * bus Bus where SuperHyway module resides.

Description

This is responsible for adding a new SuperHyway module. This sets up a new struct superhyway_device for the module being added if **sdev** == NULL.

Devices are initially added in the order that they are scanned (from the top-down of the memory map), and are assigned an ID based on the order that they are added. Any manual addition of a module will thus get the ID after the devices already discovered regardless of where it resides in memory.

Further work can and should be done in `superhyway_scan_bus()`, to be sure that any new modules are properly discovered and subsequently registered.

int **superhyway_register_driver**(struct superhyway_driver * *drv*)
Register a new SuperHyway driver

Parameters

struct superhyway_driver * drv SuperHyway driver to register.

Description

This registers the passed in **drv**. Any devices matching the id table will automatically be populated and handed off to the driver's specified probe routine.

void **superhyway_unregister_driver**(struct superhyway_driver * *drv*)
Unregister a SuperHyway driver

Parameters

struct superhyway_driver * drv SuperHyway driver to unregister.

Description

This cleans up after `superhyway_register_driver()`, and should be invoked in the exit path of any module drivers.

Maple

int **maple_driver_register**(struct maple_driver * *drv*)
register a maple driver

Parameters

struct maple_driver * drv maple driver to be registered.

Description

Registers the passed in **drv**, while updating the bus type. Devices with matching function IDs will be automatically probed.

void **maple_driver_unregister**(struct maple_driver * *drv*)
unregister a maple driver.

Parameters

struct maple_driver * drv maple driver to unregister.

Description

Cleans up after *maple_driver_register()*. To be invoked in the exit path of any module drivers.

void **maple_getcond_callback**(struct maple_device * *dev*, void (*callback) (struct mapleq **mq*, unsigned long *interval*, unsigned long *function*)
setup handling MAPLE_COMMAND_GETCOND

Parameters

struct maple_device * dev device responding

void (*) (struct mapleq *mq) callback handler callback

unsigned long interval interval in jiffies between callbacks

unsigned long function the function code for the device

int **maple_add_packet**(struct maple_device * *mdev*, u32 *function*, u32 *command*, size_t *length*, void * *data*)
add a single instruction to the maple bus queue

Parameters

struct maple_device * mdev maple device

u32 function function on device being queried

u32 command maple command to add

size_t length length of command string (in 32 bit words)

void * data remainder of command string

Symbols

`__flush_tlb_slot` (C function), 5

A

`aica_rtc_gettimeofday` (C function), 7

`aica_rtc_settimeofday` (C function), 7

F

`for_each_dtlb_entry` (C function), 4

`for_each_itlb_entry` (C function), 5

I

`ilsel_disable` (C function), 8

`ilsel_enable` (C function), 7

`ilsel_enable_fixed` (C function), 7

M

`maple_add_packet` (C function), 10

`maple_driver_register` (C function), 10

`maple_driver_unregister` (C function), 10

`maple_getcond_callback` (C function), 10

S

`sh64_get_wired_dtlb_entry` (C function), 4

`sh64_next_free_dtlb_entry` (C function), 4

`sh64_put_wired_dtlb_entry` (C function), 4

`sh64_setup_tlb_slot` (C function), 4

`sh64_tearardown_tlb_slot` (C function), 4

`sh64_tlb_init` (C function), 4

`sq_flush_range` (C function), 3

`sq_remap` (C function), 3

`sq_unmap` (C function), 3

`superhyway_add_device` (C function), 9

`superhyway_register_driver` (C function), 9

`superhyway_unregister_driver` (C function), 9