libmetal

Generated by Doxygen 1.8.6

Tue Apr 18 2017 11:50:22

Contents

1 Main Page 1

2 Software License Agreement (BSD License) 5

3 Module Index 7

3.1 Modules . . . 7

4 Data Structure Index 9

4.1 Data Structures . . . 9

5 File Index 11

5.1 File List . . . 11

6 Module Documentation 13

6.1 Allocation Interfaces . . . 13

6.1.1 Detailed Description . . . 13

6.1.2 Function Documentation . . . 13

6.1.2.1 metal_allocate_memory . . . 13

6.1.2.2 metal_free_memory . . . 13

6.2 CACHE Interfaces . . . 14

6.2.1 Detailed Description . . . 14

6.2.2 Function Documentation . . . 14

6.2.2.1 metal_cache_flush . . . 14

6.2.2.2 metal_cache_invalidate . . . 14

6.3 Condition Variable Interfaces . . . 15

6.3.1 Detailed Description . . . 15

6.3.2 Function Documentation . . . 15

6.3.2.1 metal_condition_broadcast . . . 15

6.3.2.2 metal_condition_init . . . 15

6.3.2.3 metal_condition_signal . . . 15

6.3.2.4 metal_condition_wait . . . 16

6.4 Bus Abstraction . . . 18

6.4.1 Detailed Description . . . 18

6.4.2 Macro Definition Documentation . . . 18
6.4.2.1 METAL_MAX_DEVICE_REGIONS . . . 18
6.4.3 Function Documentation . . . 18
6.4.3.1 metal_bus_find . . . 18
6.4.3.2 metal_bus_register . . . 19
6.4.3.3 metal_bus_unregister . . . 19
6.4.3.4 metal_device_close . . . 19
6.4.3.5 metal_device_io_region . . . 19
6.4.3.6 metal_device_open . . . 19
6.4.3.7 metal_register_generic_device . . . 20
6.4.4 Variable Documentation . . . 20
6.4.4.1 metal_generic_bus . . . 20
6.5 DMA Interfaces . . . 21
6.5.1 Detailed Description . . . 21
6.5.2 Macro Definition Documentation . . . 21
6.5.2.1 METAL_DMA_DEV_R . . . 21
6.5.2.2 METAL_DMA_DEV_W . . . 21
6.5.2.3 METAL_DMA_DEV_WR . . . 21
6.5.3 Function Documentation . . . 21
6.5.3.1 metal_dma_map . . . 21
6.5.3.2 metal_dma_unmap . . . 22
6.6 IO Interfaces . . . 23
6.6.1 Detailed Description . . . 24
6.6.2 Macro Definition Documentation . . . 24
6.6.2.1 METAL_CACHE_UNKNOWN . . . 24
6.6.2.2 METAL_CACHE_WB . . . 24
6.6.2.3 METAL_CACHE_WT . . . 24
6.6.2.4 METAL_DMA_CACHE_OPS . . . 24
6.6.2.5 METAL_DMA_NO_CACHE_OPS . . . 24
6.6.2.6 METAL_IO_MAPPED . . . 25
6.6.2.7 metal_io_read16 . . . 25
6.6.2.8 metal_io_read16_explicit . . . 25
6.6.2.9 metal_io_read32 . . . 25
6.6.2.10 metal_io_read32_explicit . . . 25
6.6.2.11 metal_io_read64 . . . 25
6.6.2.12 metal_io_read64_explicit . . . 25
6.6.2.13 metal_io_read8 . . . 25
6.6.2.14 metal_io_read8_explicit . . . 25
6.6.2.15 metal_io_write16 . . . 25
6.6.2.16 metal_io_write16_explicit . . . 25

6.6.2.17 metal_io_write32 . . . 25
6.6.2.18 metal_io_write32_explicit . . . 25
6.6.2.19 metal_io_write64 . . . 25
6.6.2.20 metal_io_write64_explicit . . . 25
6.6.2.21 metal_io_write8 . . . 25
6.6.2.22 metal_io_write8_explicit . . . 25
6.6.2.23 METAL_MEM_MAPPED . . . 25
6.6.2.24 METAL_SHARED_MEM . . . 25
6.6.2.25 METAL_UNCACHED . . . 25
6.6.3 Function Documentation . . . 25
6.6.3.1 metal_io_finish . . . 25
6.6.3.2 metal_io_init . . . 26
6.6.3.3 metal_io_mem_map . . . 27
6.6.3.4 metal_io_phys . . . 27
6.6.3.5 metal_io_phys_to_offset . . . 27
6.6.3.6 metal_io_phys_to_virt . . . 27
6.6.3.7 metal_io_read . . . 28
6.6.3.8 metal_io_region_size . . . 28
6.6.3.9 metal_io_virt . . . 28
6.6.3.10 metal_io_virt_to_offset . . . 28
6.6.3.11 metal_io_virt_to_phys . . . 29
6.6.3.12 metal_io_write . . . 29
6.6.3.13 metal_memcpy_io . . . 29
6.6.3.14 metal_memset_io . . . 30
6.6.4 Variable Documentation . . . 31
6.6.4.1 close . . . 31
6.6.4.2 mem_flags . . . 31
6.6.4.3 ops . . . 31
6.6.4.4 page_mask . . . 31
6.6.4.5 page_shift . . . 31
6.6.4.6 physmap . . . 31
6.6.4.7 read . . . 31
6.6.4.8 size . . . 31
6.6.4.9 virt . . . 31
6.6.4.10 write . . . 31
6.7 Interrupt Handling Interfaces . . . 32
6.7.1 Detailed Description . . . 32
6.7.2 Macro Definition Documentation . . . 32
6.7.2.1 METAL_IRQ_HANDLED . . . 32
6.7.2.2 METAL_IRQ_NOT_HANDLED . . . 32

6.7.3 Typedef Documentation . . . 32
6.7.3.1 metal_irq_handler . . . 32
6.7.4 Function Documentation . . . 33
6.7.4.1 metal_irq_disable . . . 33
6.7.4.2 metal_irq_enable . . . 33
6.7.4.3 metal_irq_register . . . 33
6.7.4.4 metal_irq_restore_enable . . . 33
6.7.4.5 metal_irq_save_disable . . . 33
6.8 List Primitives . . . 34
6.8.1 Detailed Description . . . 34
6.8.2 Macro Definition Documentation . . . 34
6.8.2.1 METAL_DECLARE_LIST . . . 34
6.8.2.2 metal_list_for_each . . . 34
6.8.3 Function Documentation . . . 34
6.8.3.1 metal_list_add_after . . . 34
6.8.3.2 metal_list_add_before . . . 34
6.8.3.3 metal_list_add_head . . . 34
6.8.3.4 metal_list_add_tail . . . 34
6.8.3.5 metal_list_del . . . 34
6.8.3.6 metal_list_first . . . 34
6.8.3.7 metal_list_init . . . 34
6.8.3.8 metal_list_is_empty . . . 35
6.9 Library Logging Interfaces . . . 36
6.9.1 Detailed Description . . . 36
6.9.2 Macro Definition Documentation . . . 36
6.9.2.1 metal_log . . . 36
6.9.3 Typedef Documentation . . . 36
6.9.3.1 metal_log_handler . . . 36
6.9.4 Enumeration Type Documentation . . . 37
6.9.4.1 metal_log_level . . . 37
6.9.5 Function Documentation . . . 37
6.9.5.1 metal_default_log_handler . . . 37
6.9.5.2 metal_get_log_handler . . . 37
6.9.5.3 metal_get_log_level . . . 37
6.9.5.4 metal_set_log_handler . . . 37
6.9.5.5 metal_set_log_level . . . 38
6.10 Mutex Interfaces . . . 39
6.10.1 Detailed Description . . . 39
6.10.2 Function Documentation . . . 39
6.10.2.1 metal_mutex_acquire . . . 39

6.10.2.2 metal_mutex_deinit . . . 39
6.10.2.3 metal_mutex_init . . . 39
6.10.2.4 metal_mutex_is_acquired . . . 39
6.10.2.5 metal_mutex_release . . . 40
6.10.2.6 metal_mutex_try_acquire . . . 40
6.11 Shared Memory Interfaces . . . 41
6.11.1 Detailed Description . . . 41
6.11.2 Function Documentation . . . 41
6.11.2.1 metal_shmem_open . . . 41
6.11.2.2 metal_shmem_register_generic . . . 41
6.12 Sleep Interfaces . . . 42
6.12.1 Detailed Description . . . 42
6.12.2 Function Documentation . . . 42
6.12.2.1 metal_sleep_usec . . . 42
6.13 Spinlock Interfaces . . . 43
6.13.1 Detailed Description . . . 43
6.13.2 Macro Definition Documentation . . . 43
6.13.2.1 METAL_SPINLOCK_INIT . . . 43
6.13.3 Function Documentation . . . 43
6.13.3.1 metal_spinlock_acquire . . . 43
6.13.3.2 metal_spinlock_init . . . 43
6.13.3.3 metal_spinlock_release . . . 43
6.14 Top Level Interfaces . . . 45
6.14.1 Detailed Description . . . 45
6.14.2 Macro Definition Documentation . . . 45
6.14.2.1 METAL_BAD_IRQ . . . 45
6.14.2.2 METAL_BAD_OFFSET . . . 45
6.14.2.3 METAL_BAD_PHYS . . . 45
6.14.2.4 METAL_INIT_DEFAULTS . . . 46
6.14.3 Typedef Documentation . . . 46
6.14.3.1 metal_irq_t . . . 46
6.14.3.2 metal_phys_addr_t . . . 46
6.14.4 Function Documentation . . . 46
6.14.4.1 metal_finish . . . 46
6.14.4.2 metal_init . . . 46
6.14.5 Variable Documentation . . . 46
6.14.5.1 _metal . . . 46
6.15 TIME Interfaces . . . 47
6.15.1 Detailed Description . . . 47
6.15.2 Function Documentation . . . 47

6.15.2.1 metal_get_timestamp . . . 47
6.16 Simple Utilities . . . 48
6.16.1 Detailed Description . . . 48
6.16.2 Macro Definition Documentation . . . 48
6.16.2.1 metal_align_down . . . 48
6.16.2.2 metal_align_up . . . 48
6.16.2.3 metal_bit . . . 48
6.16.2.4 metal_bitmap_for_each_clear_bit . . . 48
6.16.2.5 metal_bitmap_for_each_set_bit . . . 49
6.16.2.6 metal_bitmap_longs . . . 49
6.16.2.7 METAL_BITS_PER_ULONG . . . 49
6.16.2.8 metal_container_of . . . 49
6.16.2.9 metal_dim . . . 49
6.16.2.10 metal_div_round_down . . . 49
6.16.2.11 metal_div_round_up . . . 49
6.16.2.12 metal_max . . . 49
6.16.2.13 metal_min . . . 49
6.16.2.14 metal_offset_of . . . 49
6.16.2.15 metal_ptr_align_down . . . 49
6.16.2.16 metal_ptr_align_up . . . 49
6.16.2.17 metal_sign . . . 50
6.16.2.18 metal_unused . . . 50
6.16.3 Function Documentation . . . 50
6.16.3.1 metal_bitmap_clear_bit . . . 50
6.16.3.2 metal_bitmap_is_bit_clear . . . 50
6.16.3.3 metal_bitmap_is_bit_set . . . 50
6.16.3.4 metal_bitmap_next_clear_bit . . . 50
6.16.3.5 metal_bitmap_next_set_bit . . . 50
6.16.3.6 metal_bitmap_set_bit . . . 50
6.16.3.7 metal_log2 . . . 50
6.17 Library Version Interfaces . . . 51
6.17.1 Detailed Description . . . 51
6.17.2 Function Documentation . . . 51
6.17.2.1 metal_ver . . . 51
6.17.2.2 metal_ver_major . . . 51
6.17.2.3 metal_ver_minor . . . 51
6.17.2.4 metal_ver_patch . . . 52

7 Data Structure Documentation **53**

7.1 linux_bus Struct Reference . . . 53

7.1.1 Field Documentation . . . 53
7.1.1.1 bus . . . 53
7.1.1.2 bus_name . . . 53
7.1.1.3 drivers . . . 53
7.1.1.4 sbus . . . 53
7.2 linux_device Struct Reference . . . 53
7.2.1 Field Documentation . . . 54
7.2.1.1 cls_path . . . 54
7.2.1.2 dev_name . . . 54
7.2.1.3 dev_path . . . 54
7.2.1.4 device . . . 54
7.2.1.5 fd . . . 54
7.2.1.6 ldrv . . . 54
7.2.1.7 override . . . 54
7.2.1.8 region_phys . . . 54
7.2.1.9 sdev . . . 54
7.3 linux_driver Struct Reference . . . 54
7.3.1 Field Documentation . . . 54
7.3.1.1 cls_name . . . 54
7.3.1.2 dev_close . . . 54
7.3.1.3 dev_dma_map . . . 54
7.3.1.4 dev_dma_unmap . . . 54
7.3.1.5 dev_irq_ack . . . 54
7.3.1.6 dev_open . . . 55
7.3.1.7 drv_name . . . 55
7.3.1.8 mod_name . . . 55
7.3.1.9 sdrv . . . 55
7.4 metal_bus Struct Reference . . . 55
7.4.1 Detailed Description . . . 55
7.4.2 Field Documentation . . . 55
7.4.2.1 devices . . . 55
7.4.2.2 name . . . 55
7.4.2.3 node . . . 55
7.4.2.4 ops . . . 55
7.5 metal_bus_ops Struct Reference . . . 55
7.5.1 Detailed Description . . . 56
7.5.2 Field Documentation . . . 56
7.5.2.1 bus_close . . . 56
7.5.2.2 dev_close . . . 56
7.5.2.3 dev_dma_map . . . 56

7.5.2.4 dev_dma_unmap . . . 56
7.5.2.5 dev_irq_ack . . . 56
7.5.2.6 dev_open . . . 56
7.6 metal_common_state Struct Reference . . . 56
7.6.1 Detailed Description . . . 56
7.6.2 Field Documentation . . . 57
7.6.2.1 bus_list . . . 57
7.6.2.2 generic_device_list . . . 57
7.6.2.3 generic_shmem_list . . . 57
7.6.2.4 log_handler . . . 57
7.6.2.5 log_level . . . 57
7.7 metal_condition Struct Reference . . . 57
7.7.1 Field Documentation . . . 57
7.7.1.1 m . . . 57
7.7.1.2 v . . . 57
7.7.1.3 waiters . . . 57
7.7.1.4 wakeups . . . 58
7.8 metal_device Struct Reference . . . 58
7.8.1 Detailed Description . . . 58
7.8.2 Field Documentation . . . 58
7.8.2.1 bus . . . 58
7.8.2.2 irq_info . . . 58
7.8.2.3 irq_num . . . 58
7.8.2.4 name . . . 58
7.8.2.5 node . . . 58
7.8.2.6 num_regions . . . 58
7.8.2.7 regions . . . 58
7.9 metal_generic_shmem Struct Reference . . . 58
7.9.1 Detailed Description . . . 59
7.9.2 Field Documentation . . . 59
7.9.2.1 io . . . 59
7.9.2.2 name . . . 59
7.9.2.3 node . . . 59
7.10 metal_init_params Struct Reference . . . 59
7.10.1 Detailed Description . . . 59
7.10.2 Field Documentation . . . 59
7.10.2.1 log_handler . . . 59
7.10.2.2 log_level . . . 59
7.11 metal_io_ops Struct Reference . . . 60
7.11.1 Detailed Description . . . 60

7.12 metal_io_region Struct Reference . . . 60
7.12.1 Detailed Description . . . 60
7.13 metal_irq_hddesc Struct Reference . . . 60
7.13.1 Detailed Description . . . 60
7.13.2 Field Documentation . . . 61
7.13.2.1 dev . . . 61
7.13.2.2 drv_id . . . 61
7.13.2.3 hd . . . 61
7.14 metal_irqs_state Struct Reference . . . 61
7.14.1 Field Documentation . . . 61
7.14.1.1 hds . . . 61
7.14.1.2 intr_enable . . . 61
7.14.1.3 irq_lock . . . 61
7.14.1.4 irq_pthread . . . 61
7.14.1.5 irq_reg_fd . . . 61
7.14.1.6 irq_reg_stat . . . 62
7.14.1.7 irq_state . . . 62
7.15 metal_list Struct Reference . . . 62
7.15.1 Field Documentation . . . 62
7.15.1.1 next . . . 62
7.15.1.2 prev . . . 62
7.16 metal_mutex_t Struct Reference . . . 62
7.16.1 Field Documentation . . . 62
7.16.1.1 m . . . 62
7.16.1.2 v . . . 62
7.17 metal_page_size Struct Reference . . . 63
7.17.1 Detailed Description . . . 63
7.17.2 Field Documentation . . . 63
7.17.2.1 mmap_flags . . . 63
7.17.2.2 page_shift . . . 63
7.17.2.3 page_size . . . 63
7.17.2.4 path . . . 63
7.18 metal_sg Struct Reference . . . 63
7.18.1 Detailed Description . . . 64
7.18.2 Field Documentation . . . 64
7.18.2.1 io . . . 64
7.18.2.2 len . . . 64
7.18.2.3 virt . . . 64
7.19 metal_shmem Struct Reference . . . 64
7.19.1 Field Documentation . . . 64

7.19.1.1 io . . . 64
7.19.1.2 phys . . . 64
7.20 metal_spinlock Struct Reference . . . 64
7.20.1 Field Documentation . . . 65
7.20.1.1 v . . . 65
7.21 metal_state Struct Reference . . . 65
7.21.1 Detailed Description . . . 65
7.21.2 Field Documentation . . . 65
7.21.2.1 common . . . 65
7.21.2.2 data_fd . . . 65
7.21.2.3 num_page_sizes . . . 65
7.21.2.4 page_shift . . . 65
7.21.2.5 page_size . . . 66
7.21.2.6 page_sizes . . . 66
7.21.2.7 pagemap_fd . . . 66
7.21.2.8 sysfs_path . . . 66
7.21.2.9 tmp_path . . . 66

8 File Documentation **67**
8.1 lib/alloc.h File Reference . . . 67
8.2 lib/system/freertos/alloc.h File Reference . . . 67
8.2.1 Function Documentation . . . 67
8.2.1.1 metal_allocate_memory . . . 67
8.2.1.2 metal_free_memory . . . 67
8.3 lib/system/generic/alloc.h File Reference . . . 67
8.3.1 Function Documentation . . . 68
8.3.1.1 metal_allocate_memory . . . 68
8.3.1.2 metal_free_memory . . . 68
8.4 lib/system/linux/alloc.h File Reference . . . 68
8.4.1 Function Documentation . . . 68
8.4.1.1 metal_allocate_memory . . . 68
8.4.1.2 metal_free_memory . . . 68
8.5 lib/atomic.h File Reference . . . 68
8.6 lib/compiler/gcc/atomic.h File Reference . . . 68
8.6.1 Macro Definition Documentation . . . 69
8.6.1.1 atomic_compare_exchange_strong . . . 69
8.6.1.2 atomic_compare_exchange_strong_explicit . . . 70
8.6.1.3 atomic_compare_exchange_weak . . . 70
8.6.1.4 atomic_compare_exchange_weak_explicit . . . 70
8.6.1.5 atomic_exchange . . . 70

8.6.1.6 atomic_exchange_explicit . . . 70
8.6.1.7 atomic_fetch_add . . . 70
8.6.1.8 atomic_fetch_add_explicit . . . 70
8.6.1.9 atomic_fetch_and . . . 70
8.6.1.10 atomic_fetch_and_explicit . . . 70
8.6.1.11 atomic_fetch_or . . . 70
8.6.1.12 atomic_fetch_or_explicit . . . 70
8.6.1.13 atomic_fetch_sub . . . 70
8.6.1.14 atomic_fetch_sub_explicit . . . 70
8.6.1.15 atomic_fetch_xor . . . 70
8.6.1.16 atomic_fetch_xor_explicit . . . 70
8.6.1.17 atomic_flag_clear . . . 70
8.6.1.18 atomic_flag_clear_explicit . . . 70
8.6.1.19 ATOMIC_FLAG_INIT . . . 70
8.6.1.20 atomic_flag_test_and_set . . . 70
8.6.1.21 atomic_flag_test_and_set_explicit . . . 70
8.6.1.22 atomic_init . . . 70
8.6.1.23 atomic_is_lock_free . . . 71
8.6.1.24 atomic_load . . . 71
8.6.1.25 atomic_load_explicit . . . 71
8.6.1.26 atomic_signal_fence . . . 71
8.6.1.27 atomic_store . . . 71
8.6.1.28 atomic_store_explicit . . . 71
8.6.1.29 atomic_thread_fence . . . 71
8.6.1.30 ATOMIC_VAR_INIT . . . 71
8.6.2 Typedef Documentation . . . 71
8.6.2.1 atomic_char . . . 71
8.6.2.2 atomic_flag . . . 71
8.6.2.3 atomic_int . . . 71
8.6.2.4 atomic_llong . . . 71
8.6.2.5 atomic_long . . . 71
8.6.2.6 atomic_short . . . 71
8.6.2.7 atomic_uchar . . . 71
8.6.2.8 atomic_uint . . . 71
8.6.2.9 atomic_ullong . . . 71
8.6.2.10 atomic_ulong . . . 71
8.6.2.11 atomic_ushort . . . 71
8.6.3 Enumeration Type Documentation . . . 71
8.6.3.1 memory_order . . . 71
8.7 lib/processor/aarch64/atomic.h File Reference . . . 71

8.8 lib/processor/arm/atomic.h File Reference . . . 72
8.9 lib/processor/x86_64/atomic.h File Reference . . . 72
8.10 lib/cache.h File Reference . . . 72
8.11 lib/compiler/gcc/compiler.h File Reference . . . 72
8.11.1 Macro Definition Documentation . . . 72
8.11.1.1 metal_align . . . 72
8.12 lib/compiler.h File Reference . . . 72
8.13 lib/condition.h File Reference . . . 72
8.14 lib/system/freertos/condition.h File Reference . . . 72
8.14.1 Macro Definition Documentation . . . 73
8.14.1.1 METAL_CONDITION_INIT . . . 73
8.14.2 Function Documentation . . . 73
8.14.2.1 metal_condition_broadcast . . . 73
8.14.2.2 metal_condition_init . . . 73
8.14.2.3 metal_condition_signal . . . 73
8.15 lib/system/generic/condition.h File Reference . . . 73
8.15.1 Macro Definition Documentation . . . 74
8.15.1.1 METAL_CONDITION_INIT . . . 74
8.15.2 Function Documentation . . . 74
8.15.2.1 metal_condition_broadcast . . . 74
8.15.2.2 metal_condition_init . . . 74
8.15.2.3 metal_condition_signal . . . 74
8.16 lib/system/linux/condition.h File Reference . . . 74
8.16.1 Macro Definition Documentation . . . 74
8.16.1.1 METAL_CONDITION_INIT . . . 74
8.16.2 Function Documentation . . . 74
8.16.2.1 metal_condition_broadcast . . . 74
8.16.2.2 metal_condition_init . . . 75
8.16.2.3 metal_condition_signal . . . 75
8.17 lib/config.h File Reference . . . 75
8.17.1 Macro Definition Documentation . . . 75
8.17.1.1 HAVE_FUTEX_H . . . 75
8.17.1.2 HAVE_STDATOMIC_H . . . 75
8.17.1.3 METAL_MACHINE . . . 75
8.17.1.4 METAL_MACHINE_ . . . 75
8.17.1.5 METAL_PROCESSOR . . . 75
8.17.1.6 METAL_PROCESSOR_ . . . 75
8.17.1.7 METAL_SYSTEM . . . 75
8.17.1.8 METAL_SYSTEM_ . . . 75
8.17.1.9 METAL_VER . . . 75

8.17.1.10 METAL_VER_MAJOR . . . 75
8.17.1.11 METAL_VER_MINOR . . . 76
8.17.1.12 METAL_VER_PATCH . . . 76
8.18 lib/cpu.h File Reference . . . 76
8.19 lib/processor/aarch64/cpu.h File Reference . . . 76
8.19.1 Macro Definition Documentation . . . 76
8.19.1.1 metal_cpu_yield . . . 76
8.20 lib/processor/arm/cpu.h File Reference . . . 76
8.20.1 Macro Definition Documentation . . . 76
8.20.1.1 metal_cpu_yield . . . 76
8.21 lib/processor/x86_64/cpu.h File Reference . . . 76
8.21.1 Macro Definition Documentation . . . 76
8.21.1.1 metal_cpu_yield . . . 76
8.22 lib/device.c File Reference . . . 77
8.22.1 Function Documentation . . . 77
8.22.1.1 metal_generic_dev_dma_map . . . 77
8.22.1.2 metal_generic_dev_dma_unmap . . . 77
8.22.1.3 metal_generic_dev_open . . . 77
8.23 lib/system/linux/device.c File Reference . . . 78
8.23.1 Macro Definition Documentation . . . 79
8.23.1.1 for_each_linux_bus . . . 79
8.23.1.2 for_each_linux_driver . . . 79
8.23.1.3 MAX_DRIVERS . . . 79
8.23.2 Function Documentation . . . 79
8.23.2.1 metal_linux_bus_close . . . 79
8.23.2.2 metal_linux_bus_finish . . . 79
8.23.2.3 metal_linux_bus_init . . . 79
8.23.2.4 metal_linux_dev_close . . . 79
8.23.2.5 metal_linux_dev_dma_map . . . 79
8.23.2.6 metal_linux_dev_dma_unmap . . . 79
8.23.2.7 metal_linux_dev_irq_ack . . . 79
8.23.2.8 metal_linux_dev_open . . . 79
8.23.2.9 metal_linux_probe_bus . . . 79
8.23.2.10 metal_linux_probe_driver . . . 79
8.23.2.11 metal_linux_register_bus . . . 79
8.23.2.12 metal_uio_dev_bind . . . 79
8.23.2.13 metal_uio_dev_close . . . 79
8.23.2.14 metal_uio_dev_dma_map . . . 79
8.23.2.15 metal_uio_dev_dma_unmap . . . 79
8.23.2.16 metal_uio_dev_irq_ack . . . 79

8.23.2.17 metal_uio_dev_open . . . 79
8.23.2.18 metal_uio_read_map_attr . . . 79
8.23.2.19 to_linux_bus . . . 79
8.23.2.20 to_linux_device . . . 79
8.23.3 Variable Documentation . . . 80
8.23.3.1 linux_bus . . . 80
8.23.3.2 metal_linux_bus_ops . . . 80
8.24 lib/device.h File Reference . . . 80
8.25 lib/dma.c File Reference . . . 81
8.26 lib/dma.h File Reference . . . 81
8.27 lib/init.c File Reference . . . 82
8.28 lib/system/freertos/init.c File Reference . . . 82
8.28.1 Function Documentation . . . 82
8.28.1.1 metal_irq_deinit . . . 82
8.28.1.2 metal_irq_init . . . 82
8.28.1.3 metal_sys_finish . . . 82
8.28.1.4 metal_sys_init . . . 82
8.29 lib/system/generic/init.c File Reference . . . 82
8.29.1 Function Documentation . . . 83
8.29.1.1 metal_irq_deinit . . . 83
8.29.1.2 metal_irq_init . . . 83
8.29.1.3 metal_sys_finish . . . 83
8.29.1.4 metal_sys_init . . . 83
8.30 lib/system/linux/init.c File Reference . . . 83
8.30.1 Function Documentation . . . 83
8.30.1.1 metal_add_page_size . . . 83
8.30.1.2 metal_init_page_sizes . . . 83
8.30.1.3 metal_linux_irq_init . . . 83
8.30.1.4 metal_linux_irq_shutdown . . . 84
8.30.1.5 metal_pagesize_compare . . . 84
8.30.1.6 metal_sys_finish . . . 84
8.30.1.7 metal_sys_init . . . 84
8.31 lib/io.c File Reference . . . 84
8.31.1 Function Documentation . . . 84
8.31.1.1 metal_generic_memcpy_io . . . 84
8.31.1.2 metal_generic_memset_io . . . 84
8.32 lib/system/freertos/io.c File Reference . . . 84
8.32.1 Function Documentation . . . 84
8.32.1.1 metal_machine_io_mem_map . . . 84
8.33 lib/system/generic/io.c File Reference . . . 85

8.33.1 Function Documentation . . . 85
8.33.1.1 metal_machine_io_mem_map . . . 85
8.34 lib/system/linux/io.c File Reference . . . 85
8.35 lib/io.h File Reference . . . 85
8.36 lib/irq.h File Reference . . . 87
8.37 lib/system/freertos/irq.h File Reference . . . 87
8.37.1 Function Documentation . . . 87
8.37.1.1 metal_irq_isr . . . 87
8.38 lib/system/generic/irq.h File Reference . . . 88
8.38.1 Function Documentation . . . 88
8.38.1.1 metal_irq_isr . . . 88
8.39 lib/system/linux/irq.h File Reference . . . 88
8.40 lib/list.h File Reference . . . 88
8.41 lib/log.c File Reference . . . 89
8.42 lib/log.h File Reference . . . 89
8.43 lib/mutex.h File Reference . . . 90
8.44 lib/system/freertos/mutex.h File Reference . . . 90
8.44.1 Macro Definition Documentation . . . 91
8.44.1.1 METAL_MUTEX_INIT . . . 91
8.44.2 Function Documentation . . . 91
8.44.2.1 metal_mutex_acquire . . . 91
8.44.2.2 metal_mutex_deinit . . . 91
8.44.2.3 metal_mutex_init . . . 91
8.44.2.4 metal_mutex_is_acquired . . . 91
8.44.2.5 metal_mutex_release . . . 91
8.44.2.6 metal_mutex_try_acquire . . . 91
8.45 lib/system/generic/mutex.h File Reference . . . 91
8.45.1 Macro Definition Documentation . . . 91
8.45.1.1 METAL_MUTEX_INIT . . . 91
8.45.2 Function Documentation . . . 91
8.45.2.1 metal_mutex_acquire . . . 91
8.45.2.2 metal_mutex_deinit . . . 91
8.45.2.3 metal_mutex_init . . . 91
8.45.2.4 metal_mutex_is_acquired . . . 92
8.45.2.5 metal_mutex_release . . . 92
8.45.2.6 metal_mutex_try_acquire . . . 92
8.46 lib/system/linux/mutex.h File Reference . . . 92
8.46.1 Macro Definition Documentation . . . 92
8.46.1.1 METAL_MUTEX_INIT . . . 92
8.46.2 Function Documentation . . . 92

8.46.2.1 __metal_mutex_cmpxchg . . . 92
8.46.2.2 metal_mutex_acquire . . . 92
8.46.2.3 metal_mutex_deinit . . . 92
8.46.2.4 metal_mutex_init . . . 92
8.46.2.5 metal_mutex_is_acquired . . . 92
8.46.2.6 metal_mutex_release . . . 92
8.46.2.7 metal_mutex_try_acquire . . . 92
8.47 lib/shmem.c File Reference . . . 93
8.47.1 Function Documentation . . . 93
8.47.1.1 metal_shmem_open_generic . . . 93
8.48 lib/system/freertos/shmem.c File Reference . . . 93
8.49 lib/system/generic/shmem.c File Reference . . . 93
8.50 lib/system/linux/shmem.c File Reference . . . 93
8.50.1 Function Documentation . . . 94
8.50.1.1 metal_shmem_io_close . . . 94
8.50.1.2 metal_shmem_try_map . . . 94
8.50.2 Variable Documentation . . . 94
8.50.2.1 metal_shmem_io_ops . . . 94
8.51 lib/shmem.h File Reference . . . 94
8.52 lib/sleep.h File Reference . . . 94
8.53 lib/spinlock.h File Reference . . . 95
8.54 lib/sys.h File Reference . . . 95
8.55 lib/system/freertos/sys.h File Reference . . . 96
8.55.1 Macro Definition Documentation . . . 96
8.55.1.1 METAL_MAX_DEVICE_REGIONS . . . 96
8.56 lib/system/freertos/zynq7/sys.h File Reference . . . 96
8.56.1 Macro Definition Documentation . . . 96
8.56.1.1 MAX_IRQS . . . 96
8.56.2 Function Documentation . . . 97
8.56.2.1 sys_irq_disable . . . 97
8.56.2.2 sys_irq_enable . . . 97
8.57 lib/system/freertos/zynqmp_r5/sys.h File Reference . . . 97
8.57.1 Macro Definition Documentation . . . 97
8.57.1.1 MAX_IRQS . . . 97
8.57.2 Function Documentation . . . 97
8.57.2.1 sys_irq_disable . . . 97
8.57.2.2 sys_irq_enable . . . 97
8.58 lib/system/generic/sys.h File Reference . . . 97
8.58.1 Macro Definition Documentation . . . 98
8.58.1.1 METAL_MAX_DEVICE_REGIONS . . . 98

8.59 lib/system/generic/zynq7/sys.h File Reference . . . 98
8.59.1 Macro Definition Documentation . . . 98
8.59.1.1 MAX_IRQS . . . 98
8.59.2 Function Documentation . . . 98
8.59.2.1 sys_irq_disable . . . 98
8.59.2.2 sys_irq_enable . . . 98
8.60 lib/system/generic/zynqmp_r5/sys.h File Reference . . . 98
8.60.1 Macro Definition Documentation . . . 98
8.60.1.1 MAX_IRQS . . . 98
8.60.2 Function Documentation . . . 99
8.60.2.1 sys_irq_disable . . . 99
8.60.2.2 sys_irq_enable . . . 99
8.61 lib/system/linux/sys.h File Reference . . . 99
8.61.1 Macro Definition Documentation . . . 99
8.61.1.1 MAX_PAGE_SIZES . . . 99
8.61.1.2 METAL_INVALID_VADDR . . . 99
8.62 lib/system/freertos/cache.c File Reference . . . 99
8.62.1 Function Documentation . . . 100
8.62.1.1 metal_machine_cache_flush . . . 100
8.62.1.2 metal_machine_cache_invalidate . . . 100
8.63 lib/system/generic/cache.c File Reference . . . 100
8.63.1 Function Documentation . . . 100
8.63.1.1 metal_machine_cache_flush . . . 100
8.63.1.2 metal_machine_cache_invalidate . . . 100
8.64 lib/system/linux/cache.c File Reference . . . 100
8.65 lib/system/freertos/condition.c File Reference . . . 100
8.66 lib/system/generic/condition.c File Reference . . . 101
8.66.1 Function Documentation . . . 101
8.66.1.1 metal_generic_default_poll . . . 101
8.67 lib/system/linux/condition.c File Reference . . . 101
8.68 lib/system/freertos/irq.c File Reference . . . 101
8.68.1 Macro Definition Documentation . . . 102
8.68.1.1 MAX_HDS . . . 102
8.68.2 Function Documentation . . . 102
8.68.2.1 metal_irq_deinit . . . 102
8.68.2.2 metal_irq_init . . . 102
8.68.2.3 metal_irq_isr . . . 102
8.68.3 Variable Documentation . . . 102
8.68.3.1 _irqs . . . 102
8.69 lib/system/generic/irq.c File Reference . . . 102

8.69.1 Macro Definition Documentation . . . 103
8.69.1.1 MAX_HDS . . . 103
8.69.2 Function Documentation . . . 103
8.69.2.1 metal_irq_deinit . . . 103
8.69.2.2 metal_irq_init . . . 103
8.69.2.3 metal_irq_isr . . . 103
8.69.3 Variable Documentation . . . 104
8.69.3.1 _irqs . . . 104
8.70 lib/system/linux/irq.c File Reference . . . 104
8.70.1 Detailed Description . . . 105
8.70.2 Macro Definition Documentation . . . 105
8.70.2.1 MAX_HDS . . . 105
8.70.2.2 MAX_IRQS . . . 105
8.70.2.3 METAL_IRQ_STOP . . . 105
8.70.3 Function Documentation . . . 105
8.70.3.1 metal_irq_restore_enable . . . 105
8.70.3.2 metal_linux_irq_handling . . . 105
8.70.3.3 metal_linux_irq_init . . . 105
8.70.3.4 metal_linux_irq_shutdown . . . 105
8.70.4 Variable Documentation . . . 106
8.70.4.1 _irqs . . . 106
8.71 lib/system/freertos/sleep.c File Reference . . . 106
8.72 lib/system/generic/sleep.c File Reference . . . 106
8.73 lib/system/linux/sleep.c File Reference . . . 106
8.74 lib/system/freertos/time.c File Reference . . . 106
8.75 lib/system/generic/time.c File Reference . . . 107
8.76 lib/system/linux/time.c File Reference . . . 107
8.76.1 Macro Definition Documentation . . . 107
8.76.1.1 NS_PER_S . . . 107
8.77 lib/system/freertos/zynq7/sys.c File Reference . . . 107
8.77.1 Macro Definition Documentation . . . 108
8.77.1.1 ARM_AR_MEM_TTB_DESC_ALL_ACCESS . . . 108
8.77.1.2 ARM_AR_MEM_TTB_DESC_AP_MANAGER . . . 108
8.77.1.3 ARM_AR_MEM_TTB_DESC_B . . . 108
8.77.1.4 ARM_AR_MEM_TTB_DESC_BACKWARDS . . . 108
8.77.1.5 ARM_AR_MEM_TTB_DESC_C . . . 108
8.77.1.6 ARM_AR_MEM_TTB_DESC_S . . . 108
8.77.1.7 ARM_AR_MEM_TTB_DESC_SECT . . . 108
8.77.1.8 ARM_AR_MEM_TTB_DESC_TEX . . . 108
8.77.1.9 ARM_AR_MEM_TTB_SECT_SIZE . . . 108

8.77.1.10 ARM_AR_MEM_TTB_SECT_SIZE_MASK . . . 109
8.77.1.11 ARM_AR_MEM_TTB_SECT_TO_DESC_SHIFT . . . 109
8.77.1.12 ARM_AR_MEM_TTB_SIZE . . . 109
8.77.1.13 ESAL_GE_MEM_32BIT_SET . . . 109
8.77.2 Function Documentation . . . 109
8.77.2.1 __attribute__ . . . 109
8.77.2.2 metal_machine_cache_flush . . . 109
8.77.2.3 metal_machine_cache_invalidate . . . 109
8.77.2.4 metal_machine_io_mem_map . . . 109
8.77.2.5 sys_irq_restore_enable . . . 109
8.77.2.6 sys_irq_save_disable . . . 109
8.77.3 Variable Documentation . . . 109
8.77.3.1 int_old_val . . . 109
8.78 lib/system/freertos/zynqmp_r5/sys.c File Reference . . . 109
8.78.1 Macro Definition Documentation . . . 110
8.78.1.1 MPU_REGION_SIZE_MIN . . . 110
8.78.2 Function Documentation . . . 110
8.78.2.1 __attribute__ . . . 110
8.78.2.2 metal_machine_cache_flush . . . 110
8.78.2.3 metal_machine_cache_invalidate . . . 110
8.78.2.4 metal_machine_io_mem_map . . . 110
8.78.2.5 sys_irq_restore_enable . . . 110
8.78.2.6 sys_irq_save_disable . . . 110
8.78.3 Variable Documentation . . . 110
8.78.3.1 int_old_val . . . 110
8.79 lib/system/generic/zynq7/sys.c File Reference . . . 110
8.79.1 Macro Definition Documentation . . . 111
8.79.1.1 ARM_AR_MEM_TTB_DESC_ALL_ACCESS . . . 111
8.79.1.2 ARM_AR_MEM_TTB_DESC_AP_MANAGER . . . 111
8.79.1.3 ARM_AR_MEM_TTB_DESC_B . . . 111
8.79.1.4 ARM_AR_MEM_TTB_DESC_BACKWARDS . . . 111
8.79.1.5 ARM_AR_MEM_TTB_DESC_C . . . 111
8.79.1.6 ARM_AR_MEM_TTB_DESC_S . . . 111
8.79.1.7 ARM_AR_MEM_TTB_DESC_SECT . . . 111
8.79.1.8 ARM_AR_MEM_TTB_DESC_TEX . . . 111
8.79.1.9 ARM_AR_MEM_TTB_SECT_SIZE . . . 111
8.79.1.10 ARM_AR_MEM_TTB_SECT_SIZE_MASK . . . 111
8.79.1.11 ARM_AR_MEM_TTB_SECT_TO_DESC_SHIFT . . . 111
8.79.1.12 ARM_AR_MEM_TTB_SIZE . . . 111
8.79.1.13 ESAL_GE_MEM_32BIT_SET . . . 111

8.79.2 Function Documentation . . . 112
8.79.2.1 __attribute__ . . . 112
8.79.2.2 metal_machine_cache_flush . . . 112
8.79.2.3 metal_machine_cache_invalidate . . . 112
8.79.2.4 metal_machine_io_mem_map . . . 112
8.79.2.5 sys_irq_restore_enable . . . 112
8.79.2.6 sys_irq_save_disable . . . 112
8.79.3 Variable Documentation . . . 112
8.79.3.1 int_old_val . . . 112
8.80 lib/system/generic/zynqmp_r5/sys.c File Reference . . . 112
8.80.1 Macro Definition Documentation . . . 113
8.80.1.1 MPU_REGION_SIZE_MIN . . . 113
8.80.2 Function Documentation . . . 113
8.80.2.1 __attribute__ . . . 113
8.80.2.2 metal_machine_cache_flush . . . 113
8.80.2.3 metal_machine_cache_invalidate . . . 113
8.80.2.4 metal_machine_io_mem_map . . . 113
8.80.2.5 sys_irq_restore_enable . . . 113
8.80.2.6 sys_irq_save_disable . . . 113
8.80.3 Variable Documentation . . . 113
8.80.3.1 int_old_val . . . 113
8.81 lib/system/linux/utilities.c File Reference . . . 113
8.81.1 Function Documentation . . . 114
8.81.1.1 metal_map . . . 114
8.81.1.2 metal_mktemp . . . 114
8.81.1.3 metal_mktemp_template . . . 114
8.81.1.4 metal_mktemp_unlinked . . . 114
8.81.1.5 metal_mlock . . . 115
8.81.1.6 metal_open . . . 115
8.81.1.7 metal_open_unlinked . . . 115
8.81.1.8 metal_randomize_string . . . 115
8.81.1.9 metal_unmap . . . 116
8.81.1.10 metal_virt2phys . . . 116
8.82 lib/time.h File Reference . . . 116
8.83 lib/utilities.h File Reference . . . 116
8.84 lib/version.c File Reference . . . 117
8.85 lib/version.h File Reference . . . 117
8.86 LICENSE.md File Reference . . . 118
8.87 README.md File Reference . . . 118

Index 119

Chapter 1

Main Page

Overview

Libmetal provides common user APIs to access devices, handle device interrupts and request memory across the following operating environments:

- Linux user space (based on UIO and VFIO support in the kernel)
- RTOS (with and without virtual memory)
- Bare-metal environments

Interfaces

The following subsections give an overview of interfaces provided by libmetal.

Platform and OS Independent Utilities

These interfaces do not need to be ported across to new operating systems.

I/O

The libmetal I/O region abstraction provides access to memory mapped I/O and shared memory regions. This includes:

- primitives to read and write memory with ordering constraints, and
- ability to translate between physical and virtual addressing on systems that support virtual memory.

Log

The libmetal logging interface is used to plug log messages generated by libmetal into application specific logging mechanisms (e.g. syslog). This also provides basic message prioritization and filtering mechanisms.

List

This is a simple doubly linked list implementation used internally within libmetal, and also available for application use.

Other Utilities

The following utilities are provided in lib/utilities.h:

- Min/max, round up/down, etc.
- Bitmap operations
- Helper to compute container structure pointers
- ... and more ...

Version

The libmetal version interface allows user to get the version of the library.

Top Level Interfaces

The users will need to call two top level interfaces to use libmetal APIs:

- metal_init - initialize the libmetal resource
- metal_finish - release libmetal resource

Each system needs to have their own implementation inside libmetal for these two APIs to call:

- metal_sys_init
- metal_sys_finish

For the current release, libmetal provides Linux userspace and bare-metal implementation for metal_sys_init and metal_sys_finish.

For Linux userspace, metal_sys_init sets up a table for available shared pages, checks whether UIO/VFIO drivers are avail, and starts interrupt handling thread.

For bare-metal, metal_sys_init and metal_sys_finish just returns.

Atomics

The libmetal atomic operations API is consistent with the C11/C++11 stdatomics interface. The stdatomics interface is commonly provided by recent toolchains including GCC and LLVM/Clang. When porting to a different toolchain, it may be necessary to provide an stdatomic compatible implementation if the toolchain does not already provide one.

Alloc

libmetal provides memory allocation and release APIs.

Locking

libmetal provides the following locking APIs.

Mutex

libmetal has a generic mutex implementation which is a busy wait. It is recommended to have OS specific implementation for mutex.

The Linux userspace mutex implementation uses futex to wait for the lock and wakeup a waiter.

Condition Variable

libmetal condition variable APIs provide "wait" for user applications to wait on some condition to be met, and "signal" to indicate a particular even occurs.

Spinlock

libmetal spinlock APIs provides busy waiting mechanism to acquire a lock.

Shmem

libmetal has a generic static shared memory implementation. If your OS has a global shared memory allocation, you will need to port it for the OS.

The Linux userspace shmem implementation uses libhugetlbfs to support huge page sizes.

Bus and Device Abstraction

libmetal has a static generic implementation. If your OS has a driver model implementation, you will need to port it for the OS.

The Linux userspace abstraction binds the devices to UIO or VFIO driver. The user applications specify which device to use, e.g. bus "platform" bus, device "f8000000.slcr", and then the abstraction will check if platform UIO driver or platform VFIO driver is there. If platform VFIO driver exists, it will bind the device to the platform VFIO driver, otherwise, if UIO driver exists, it will bind the device to the platform UIO driver.

The VFIO support is not yet implemented.

Interrupt

libmetal provides APIs to register an interrupt, disable interrupts and restore interrupts.

The Linux userspace implementation will use a thread to call select() function to listen to the file descriptors of the devices to see if there is an interrupt triggered. If there is an interrupt triggered, it will call the interrupt handler registered by the user application.

Cache

libmetal provides APIs to flush and invalidate caches.

The cache APIs for Linux userspace are empty functions for now as cache operations system calls are not avaiable for all architectures.

DMA

libmetal DMA APIs provide DMA map and unmap implementation.

After calling DMA map, the DMA device will own the memory. After calling DMA unmap, the cpu will own the memory.

For Linux userspace, it only supports to use UIO device memory as DMA memory for this release.

Time

libmetal time APIs provide getting timestamp implementation.

Sleep

libmetal sleep APIs provide getting delay execution implementation.

Compiler

This API is for compiler dependent functions. For this release, there is only a GCC implementation, and compiler specific code is limited to atomic operations.

Chapter 2

Software License Agreement (BSD License)

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of Xilinx nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

Chapter 3

Module Index

3.1 Modules

Here is a list of all modules:

Allocation Interfaces . . . 13
CACHE Interfaces . . . 14
Condition Variable Interfaces . . . 15
Bus Abstraction . . . 18
DMA Interfaces . . . 21
IO Interfaces . . . 23
Interrupt Handling Interfaces . . . 32
List Primitives . . . 34
Library Logging Interfaces . . . 36
Mutex Interfaces . . . 39
Shared Memory Interfaces . . . 41
Sleep Interfaces . . . 42
Spinlock Interfaces . . . 43
Top Level Interfaces . . . 45
TIME Interfaces . . . 47
Simple Utilities . . . 48
Library Version Interfaces . . . 51

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

linux_bus . . . 53
linux_device . . . 53
linux_driver . . . 54
metal_bus . . . 55
metal_bus_ops . . . 55
metal_common_state . . . 56
metal_condition . . . 57
metal_device . . . 58
metal_generic_shmem . . . 58
metal_init_params . . . 59
metal_io_ops . . . 60
metal_io_region . . . 60
metal_irq_hddesc . . . 60
metal_irqs_state . . . 61
metal_list . . . 62
metal_mutex_t . . . 62
metal_page_size . . . 63
metal_sg
    Scatter/gather list element structure . . . 63
metal_shmem . . . 64
metal_spinlock . . . 64
metal_state . . . 65

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

lib/alloc.h . . . 67
lib/atomic.h . . . 68
lib/cache.h . . . 72
lib/compiler.h . . . 72
lib/condition.h . . . 72
lib/config.h . . . 75
lib/cpu.h . . . 76
lib/device.c . . . 77
lib/device.h . . . 80
lib/dma.c . . . 81
lib/dma.h . . . 81
lib/init.c . . . 82
lib/io.c . . . 84
lib/io.h . . . 85
lib/irq.h . . . 87
lib/list.h . . . 88
lib/log.c . . . 89
lib/log.h . . . 89
lib/mutex.h . . . 90
lib/shmem.c . . . 93
lib/shmem.h . . . 94
lib/sleep.h . . . 94
lib/spinlock.h . . . 95
lib/sys.h . . . 95
lib/time.h . . . 116
lib/utilities.h . . . 116
lib/version.c . . . 117
lib/version.h . . . 117
lib/compiler/gcc/atomic.h . . . 68
lib/compiler/gcc/compiler.h . . . 72
lib/processor/aarch64/atomic.h . . . 71
lib/processor/aarch64/cpu.h . . . 76
lib/processor/arm/atomic.h . . . 72
lib/processor/arm/cpu.h . . . 76
lib/processor/x86_64/atomic.h . . . 72
lib/processor/x86_64/cpu.h . . . 76
lib/system/freertos/alloc.h . . . 67
lib/system/freertos/cache.c . . . 99

lib/system/freertos/condition.c . . . 100
lib/system/freertos/condition.h . . . 72
lib/system/freertos/init.c . . . 82
lib/system/freertos/io.c . . . 84
lib/system/freertos/irq.c . . . 101
lib/system/freertos/irq.h . . . 87
lib/system/freertos/mutex.h . . . 90
lib/system/freertos/shmem.c . . . 93
lib/system/freertos/sleep.c . . . 106
lib/system/freertos/sys.h . . . 96
lib/system/freertos/time.c . . . 106
lib/system/freertos/zynq7/sys.c . . . 107
lib/system/freertos/zynq7/sys.h . . . 96
lib/system/freertos/zynqmp_r5/sys.c . . . 109
lib/system/freertos/zynqmp_r5/sys.h . . . 97
lib/system/generic/alloc.h . . . 67
lib/system/generic/cache.c . . . 100
lib/system/generic/condition.c . . . 101
lib/system/generic/condition.h . . . 73
lib/system/generic/init.c . . . 82
lib/system/generic/io.c . . . 85
lib/system/generic/irq.c . . . 102
lib/system/generic/irq.h . . . 88
lib/system/generic/mutex.h . . . 91
lib/system/generic/shmem.c . . . 93
lib/system/generic/sleep.c . . . 106
lib/system/generic/sys.h . . . 97
lib/system/generic/time.c . . . 107
lib/system/generic/zynq7/sys.c . . . 110
lib/system/generic/zynq7/sys.h . . . 98
lib/system/generic/zynqmp_r5/sys.c . . . 112
lib/system/generic/zynqmp_r5/sys.h . . . 98
lib/system/linux/alloc.h . . . 68
lib/system/linux/cache.c . . . 100
lib/system/linux/condition.c . . . 101
lib/system/linux/condition.h . . . 74
lib/system/linux/device.c . . . 78
lib/system/linux/init.c . . . 83
lib/system/linux/io.c . . . 85
lib/system/linux/irq.c
Linux libmetal irq definitions . . . 104
lib/system/linux/irq.h . . . 88
lib/system/linux/mutex.h . . . 92
lib/system/linux/shmem.c . . . 93
lib/system/linux/sleep.c . . . 106
lib/system/linux/sys.h . . . 99
lib/system/linux/time.c . . . 107
lib/system/linux/utilities.c . . . 113

Chapter 6

Module Documentation

6.1 Allocation Interfaces

Functions

- static void * metal_allocate_memory (unsigned int size)

  allocate requested memory size return a pointer to the allocated memory
- static void metal_free_memory (void *ptr)

  free the memory previously allocated

6.1.1 Detailed Description

6.1.2 Function Documentation

6.1.2.1 static void* metal_allocate_memory (unsigned int *size*) [inline], [static]

allocate requested memory size return a pointer to the allocated memory

Parameters

in	*size*	size in byte of requested memory

Returns

memory pointer, or 0 if it failed to allocate

6.1.2.2 static void metal_free_memory (void * *ptr*) [inline], [static]

free the memory previously allocated

Parameters

in	*ptr*	pointer to memory

6.2 CACHE Interfaces

Functions

- void metal_cache_flush (void *addr, unsigned int len)

  flush specified data cache
- void metal_cache_invalidate (void *addr, unsigned int len)

  invalidate specified data cache

6.2.1 Detailed Description

6.2.2 Function Documentation

6.2.2.1 void metal_cache_flush (void * *addr,* unsigned int *len*)

flush specified data cache

Parameters

in	*addr*	start memory logical address
in	*len*	length of memory If addr is NULL, and len is 0, It will flush the whole data cache.

Do nothing. Do not flush cache from Linux userspace.

6.2.2.2 void metal_cache_invalidate (void * *addr,* unsigned int *len*)

invalidate specified data cache

Parameters

in	*addr*	start memory logical address
in	*len*	length of memory If addr is NULL, and len is 0, It will invalidate the whole data cache.

Do nothing. Do not invalidate cache from Linux userspace.

6.3 Condition Variable Interfaces

Functions

- static void metal_condition_init (struct metal_condition *cv)

  Initialize a libmetal condition variable.
- static int metal_condition_signal (struct metal_condition *cv)

  Notify one waiter. Before calling this function, the caller should have acquired the mutex.
- static int metal_condition_broadcast (struct metal_condition *cv)

  Notify all waiters. Before calling this function, the caller should have acquired the mutex.
- int metal_condition_wait (struct metal_condition *cv, metal_mutex_t *m)

  Block until the condition variable is notified. Before calling this function, the caller should have acquired the mutex.

6.3.1 Detailed Description

6.3.2 Function Documentation

6.3.2.1 static int metal_condition_broadcast (struct metal_condition * *cv*) [inline], [static]

Notify all waiters. Before calling this function, the caller should have acquired the mutex.

Parameters

in	*cv*	condition variable

Returns

zero on no errors, non-zero on errors

See Also

metal_condition_wait, metal_condition_signal

6.3.2.2 static void metal_condition_init (struct metal_condition * *cv*) [inline], [static]

Initialize a libmetal condition variable.

Parameters

in	*cv*	condition variable to initialize.

6.3.2.3 static int metal_condition_signal (struct metal_condition * *cv*) [inline], [static]

Notify one waiter. Before calling this function, the caller should have acquired the mutex.

Parameters

in	*cv*	condition variable

Returns

zero on no errors, non-zero on errors

See Also

metal_condition_wait, metal_condition_broadcast

6.3.2.4 int metal_condition_wait (struct metal_condition * cv, metal_mutex_t * m)

Block until the condition variable is notified. Before calling this function, the caller should have acquired the mutex.

Parameters

in	*cv*	condition variable
in	*m*	mutex

Returns

0 on success, non-zero on failure.

See Also

metal_condition_signal

6.4 Bus Abstraction

Data Structures

- struct metal_bus_ops
- struct metal_bus
- struct metal_device

Macros

- #define METAL_MAX_DEVICE_REGIONS 32

Functions

- int metal_bus_register (struct metal_bus *bus)

  Register a libmetal bus.
- int metal_bus_unregister (struct metal_bus *bus)

  Unregister a libmetal bus.
- int metal_bus_find (const char *name, struct metal_bus **bus)

  Find a libmetal bus by name.
- int metal_register_generic_device (struct metal_device *device)

  Statically register a generic libmetal device.
- int metal_device_open (const char *bus_name, const char *dev_name, struct metal_device **device)

  Open a libmetal device by name.
- void metal_device_close (struct metal_device *device)

  Close a libmetal device.
- static struct metal_io_region * metal_device_io_region (struct metal_device *device, unsigned index)

  Get an I/O region accessor for a device region.

Variables

- struct metal_bus metal_generic_bus

6.4.1 Detailed Description

6.4.2 Macro Definition Documentation

6.4.2.1 #define METAL_MAX_DEVICE_REGIONS 32

6.4.3 Function Documentation

6.4.3.1 int metal_bus_find (const char * *name*, struct metal_bus ** *bus*)

Find a libmetal bus by name.

Parameters

in	*name*	Bus name.

out	*bus*	Returned bus handle.

Returns

0 on success, or -errno on failure.

6.4.3.2 int metal_bus_register (struct metal_bus * *bus*)

Register a libmetal bus.

Parameters

in	*bus*	Pre-initialized bus structure.

Returns

0 on success, or -errno on failure.

6.4.3.3 int metal_bus_unregister (struct metal_bus * *bus*)

Unregister a libmetal bus.

Parameters

in	*bus*	Pre-registered bus structure.

Returns

0 on success, or -errno on failure.

6.4.3.4 void metal_device_close (struct metal_device * *device*)

Close a libmetal device.

Parameters

in	*device*	Device handle.

6.4.3.5 static struct metal_io_region* metal_device_io_region (struct metal_device * *device,* unsigned *index*) [static]

Get an I/O region accessor for a device region.

Parameters

in	*device*	Device handle.
in	*index*	Region index.

Returns

I/O accessor handle, or NULL on failure.

6.4.3.6 int metal_device_open (const char * *bus_name,* const char * *dev_name,* struct metal_device ** *device*)

Open a libmetal device by name.

Parameters

in	*bus_name*	Bus name.
in	*dev_name*	Device name.
out	*device*	Returned device handle.

Returns

0 on success, or -errno on failure.

6.4.3.7 int metal_register_generic_device (struct metal_device * *device*)

Statically register a generic libmetal device.

Devices may be statically registered at application initialization, or may be dynamically opened via sysfs or libfdt based enumeration at runtime. This interface is used for static registration of devices. Subsequent calls to metal_device_open() look up in this list of pre-registered devices on the "generic" bus.

Parameters

in	*device*	Generic device.

Returns

0 on success, or -errno on failure.

6.4.4 Variable Documentation

6.4.4.1 struct metal_bus metal_generic_bus

Libmetal generic bus.

6.5 DMA Interfaces

Data Structures

- struct metal_sg

  scatter/gather list element structure

Macros

- #define METAL_DMA_DEV_R 1
- #define METAL_DMA_DEV_W 2
- #define METAL_DMA_DEV_WR 3

Functions

- int metal_dma_map (struct metal_device ∗dev, uint32_t dir, struct metal_sg ∗sg_in, int nents_in, struct metal_sg ∗sg_out)

  Map memory for DMA transaction. After the memory is DMA mapped, the memory should be accessed by the DMA device but not the CPU.
- void metal_dma_unmap (struct metal_device ∗dev, uint32_t dir, struct metal_sg ∗sg, int nents)

  Unmap DMA memory After the memory is DMA unmapped, the memory should be accessed by the CPU but not the DMA device.

6.5.1 Detailed Description

6.5.2 Macro Definition Documentation

6.5.2.1 #define METAL_DMA_DEV_R 1

DMA direction, device read

6.5.2.2 #define METAL_DMA_DEV_W 2

DMA direction, device write

6.5.2.3 #define METAL_DMA_DEV_WR 3

DMA direction, device read/write

6.5.3 Function Documentation

6.5.3.1 int metal_dma_map (struct metal_device ∗ *dev,* uint32_t *dir,* struct metal_sg ∗ *sg_in,* int *nents_in,* struct metal_sg ∗ *sg_out*)

Map memory for DMA transaction. After the memory is DMA mapped, the memory should be accessed by the DMA device but not the CPU.

Parameters

in	*dev*	DMA device
in	*dir*	DMA direction
in	*sg_in*	sg list of memory to map
in	*nents_in*	number of sg list entries of memory to map
out	*sg_out*	sg list of mapped memory

Returns

number of mapped sg entries, -error on failure.

6.5.3.2 void metal_dma_unmap (struct **metal_device** * *dev,* uint32_t *dir,* struct **metal_sg** * *sg,* int *nents*)

Unmap DMA memory After the memory is DMA unmapped, the memory should be accessed by the CPU but not the DMA device.

Parameters

in	*dev*	DMA device
in	*dir*	DMA direction
in	*sg*	sg list of mapped DMA memory
in	*nents*	number of sg list entries of DMA memory

6.6 IO Interfaces

Data Structures

- struct metal_io_ops
- struct metal_io_region

Macros

- #define METAL_CACHE_UNKNOWN 0x0 /∗∗ Use cache, unknown cache type ∗/
- #define METAL_UNCACHED 0x1 /∗∗ No cache ∗/
- #define METAL_CACHE_WB 0x2 /∗∗ Write back ∗/
- #define METAL_CACHE_WT 0x4 /∗∗ Write through ∗/
- #define METAL_MEM_MAPPED 0x10 /∗∗ Memory mapped ∗/
- #define METAL_IO_MAPPED 0x20 /∗∗ I/O mapped ∗/
- #define METAL_SHARED_MEM 0x40 /∗∗ Shared memory ∗/
- #define METAL_DMA_NO_CACHE_OPS 0x0 /∗∗ No cache ops in DMA transaction ∗/
- #define METAL_DMA_CACHE_OPS 0x1 /∗∗ Require cache ops in DMA transaction ∗/
- #define metal_io_read8_explicit(_io, _ofs, _order) metal_io_read((_io), (_ofs), (_order), 1)
- #define metal_io_read8(_io, _ofs) metal_io_read((_io), (_ofs), memory_order_seq_cst, 1)
- #define metal_io_write8_explicit(_io, _ofs, _val, _order) metal_io_write((_io), (_ofs), (_val), (_order), 1)
- #define metal_io_write8(_io, _ofs, _val) metal_io_write((_io), (_ofs), (_val), memory_order_seq_cst, 1)
- #define metal_io_read16_explicit(_io, _ofs, _order) metal_io_read((_io), (_ofs), (_order), 2)
- #define metal_io_read16(_io, _ofs) metal_io_read((_io), (_ofs), memory_order_seq_cst, 2)
- #define metal_io_write16_explicit(_io, _ofs, _val, _order) metal_io_write((_io), (_ofs), (_val), (_order), 2)
- #define metal_io_write16(_io, _ofs, _val) metal_io_write((_io), (_ofs), (_val), memory_order_seq_cst, 2)
- #define metal_io_read32_explicit(_io, _ofs, _order) metal_io_read((_io), (_ofs), (_order), 4)
- #define metal_io_read32(_io, _ofs) metal_io_read((_io), (_ofs), memory_order_seq_cst, 4)
- #define metal_io_write32_explicit(_io, _ofs, _val, _order) metal_io_write((_io), (_ofs), (_val), (_order), 4)
- #define metal_io_write32(_io, _ofs, _val) metal_io_write((_io), (_ofs), (_val), memory_order_seq_cst, 4)
- #define metal_io_read64_explicit(_io, _ofs, _order) metal_io_read((_io), (_ofs), (_order), 8)
- #define metal_io_read64(_io, _ofs) metal_io_read((_io), (_ofs), memory_order_seq_cst, 8)
- #define metal_io_write64_explicit(_io, _ofs, _val, _order) metal_io_write((_io), (_ofs), (_val), (_order), 8)
- #define metal_io_write64(_io, _ofs, _val) metal_io_write((_io), (_ofs), (_val), memory_order_seq_cst, 8)

Functions

- static void metal_io_init (struct metal_io_region ∗io, void ∗virt, const metal_phys_addr_t ∗physmap, size_t size, unsigned page_shift, unsigned int mem_flags, const struct metal_io_ops ∗ops)

  Open a libmetal I/O region.
- static void metal_io_finish (struct metal_io_region ∗io)

  Close a libmetal shared memory segment.
- static size_t metal_io_region_size (struct metal_io_region ∗io)

  Get size of I/O region.
- static void ∗ metal_io_virt (struct metal_io_region ∗io, unsigned long offset)

  Get virtual address for a given offset into the I/O region.
- static unsigned long metal_io_virt_to_offset (struct metal_io_region ∗io, void ∗virt)

  Convert a virtual address to offset within I/O region.
- static metal_phys_addr_t metal_io_phys (struct metal_io_region ∗io, unsigned long offset)

  Get physical address for a given offset into the I/O region.
- static unsigned long metal_io_phys_to_offset (struct metal_io_region ∗io, metal_phys_addr_t phys)

  Convert a physical address to offset within I/O region.

- static void * metal_io_phys_to_virt (struct metal_io_region *io, metal_phys_addr_t phys)

  Convert a physical address to virtual address.
- static metal_phys_addr_t metal_io_virt_to_phys (struct metal_io_region *io, void *virt)

  Convert a virtual address to physical address.
- static uint64_t metal_io_read (struct metal_io_region *io, unsigned long offset, memory_order order, int width)

  Read a value from an I/O region.
- static void metal_io_write (struct metal_io_region *io, unsigned long offset, uint64_t value, memory_order order, int width)

  Write a value into an I/O region.
- void * metal_io_mem_map (metal_phys_addr_t pa, struct metal_io_region *io, size_t size)

  libmetal memory map
- void * metal_memset_io (void *dst, int c, size_t size)

  libmetal set device memory
- void * metal_memcpy_io (void *dst, const void *src, size_t size)

  libmetal copy to target memory

Variables

- uint64_t(* metal_io_ops::read)(struct metal_io_region *io, unsigned long offset, memory_order order, int width)
- void(* metal_io_ops::write)(struct metal_io_region *io, unsigned long offset, uint64_t value, memory_order order, int width)
- void(* metal_io_ops::close)(struct metal_io_region *io)
- void * metal_io_region::virt
- const metal_phys_addr_t * metal_io_region::physmap
- size_t metal_io_region::size
- unsigned long metal_io_region::page_shift
- metal_phys_addr_t metal_io_region::page_mask
- unsigned int metal_io_region::mem_flags
- struct metal_io_ops metal_io_region::ops

6.6.1 Detailed Description

6.6.2 Macro Definition Documentation

6.6.2.1 #define METAL_CACHE_UNKNOWN 0x0 /** Use cache, unknown cache type */

I/O memory flags macros for caching scheme Cache bits

6.6.2.2 #define METAL_CACHE_WB 0x2 /** Write back */

6.6.2.3 #define METAL_CACHE_WT 0x4 /** Write through */

6.6.2.4 #define METAL_DMA_CACHE_OPS 0x1 /** Require cache ops in DMA transaction */

6.6.2.5 #define METAL_DMA_NO_CACHE_OPS 0x0 /** No cache ops in DMA transaction */

DMA cache bits

6.6.2.6 #define METAL_IO_MAPPED 0x20 /** I/O mapped */

6.6.2.7 #define metal_io_read16(*_io, _ofs*) **metal_io_read**((_io), (_ofs), **memory_order_seq_cst**, 2)

6.6.2.8 #define metal_io_read16_explicit(*_io, _ofs, _order*) **metal_io_read**((_io), (_ofs), (_order), 2)

6.6.2.9 #define metal_io_read32(*_io, _ofs*) **metal_io_read**((_io), (_ofs), **memory_order_seq_cst**, 4)

6.6.2.10 #define metal_io_read32_explicit(*_io, _ofs, _order*) **metal_io_read**((_io), (_ofs), (_order), 4)

6.6.2.11 #define metal_io_read64(*_io, _ofs*) **metal_io_read**((_io), (_ofs), **memory_order_seq_cst**, 8)

6.6.2.12 #define metal_io_read64_explicit(*_io, _ofs, _order*) **metal_io_read**((_io), (_ofs), (_order), 8)

6.6.2.13 #define metal_io_read8(*_io, _ofs*) **metal_io_read**((_io), (_ofs), **memory_order_seq_cst**, 1)

6.6.2.14 #define metal_io_read8_explicit(*_io, _ofs, _order*) **metal_io_read**((_io), (_ofs), (_order), 1)

6.6.2.15 #define metal_io_write16(*_io, _ofs, _val*) **metal_io_write**((_io), (_ofs), (_val), **memory_order_seq_cst**, 2)

6.6.2.16 #define metal_io_write16_explicit(*_io, _ofs, _val, _order*) **metal_io_write**((_io), (_ofs), (_val), (_order), 2)

6.6.2.17 #define metal_io_write32(*_io, _ofs, _val*) **metal_io_write**((_io), (_ofs), (_val), **memory_order_seq_cst**, 4)

6.6.2.18 #define metal_io_write32_explicit(*_io, _ofs, _val, _order*) **metal_io_write**((_io), (_ofs), (_val), (_order), 4)

6.6.2.19 #define metal_io_write64(*_io, _ofs, _val*) **metal_io_write**((_io), (_ofs), (_val), **memory_order_seq_cst**, 8)

6.6.2.20 #define metal_io_write64_explicit(*_io, _ofs, _val, _order*) **metal_io_write**((_io), (_ofs), (_val), (_order), 8)

6.6.2.21 #define metal_io_write8(*_io, _ofs, _val*) **metal_io_write**((_io), (_ofs), (_val), **memory_order_seq_cst**, 1)

6.6.2.22 #define metal_io_write8_explicit(*_io, _ofs, _val, _order*) **metal_io_write**((_io), (_ofs), (_val), (_order), 1)

6.6.2.23 #define METAL_MEM_MAPPED 0x10 /** Memory mapped */

Memory types

6.6.2.24 #define METAL_SHARED_MEM 0x40 /** Shared memory */

6.6.2.25 #define METAL_UNCACHED 0x1 /** No cache */

6.6.3 Function Documentation

6.6.3.1 static void metal_io_finish (struct **metal_io_region** * *io*) [inline], [static]

Close a libmetal shared memory segment.

Parameters

in	*io*	I/O region handle.

6.6.3.2 static void metal_io_init (struct metal_io_region * *io*, void * *virt*, const metal_phys_addr_t * *physmap*, size_t *size*, unsigned *page_shift*, unsigned int *mem_flags*, const struct metal_io_ops * *ops*) [inline], [static]

Open a libmetal I/O region.

Parameters

in,out	*io*	I/O region handle.
in	*virt*	Virtual address of region.
in	*physmap*	Array of physical addresses per page.
in	*size*	Size of region.
in	*page_shift*	Log2 of page size (-1 for single page).
in	*mem_flags*	Memory flags
in	*ops*	ops

6.6.3.3 void* metal_io_mem_map (metal_phys_addr_t *pa*, struct metal_io_region * *io*, size_t *size*)

libmetal memory map

This function is to enable memory mapping to the specified I/O memory.

Parameters

in	*pa*	physical memory start address
in	*io*	memory region
in	*size*	size of the memory range

Returns

logical address if suceeded, or 0 if failed.

6.6.3.4 static metal_phys_addr_t metal_io_phys (struct metal_io_region * *io*, unsigned long *offset*) [inline], [static]

Get physical address for a given offset into the I/O region.

Parameters

in	*io*	I/O region handle.
in	*offset*	Offset into shared memory segment.

Returns

METAL_BAD_PHYS if offset is out of range, or physical address of offset.

6.6.3.5 static unsigned long metal_io_phys_to_offset (struct metal_io_region * *io*, metal_phys_addr_t *phys*) [inline], [static]

Convert a physical address to offset within I/O region.

Parameters

in	*io*	I/O region handle.
in	*phys*	Physical address within segment.

Returns

METAL_BAD_OFFSET if out of range, or offset.

6.6.3.6 static void* metal_io_phys_to_virt (struct metal_io_region * *io*, metal_phys_addr_t *phys*) [inline], [static]

Convert a physical address to virtual address.

Parameters

in	*io*	Shared memory segment handle.
in	*phys*	Physical address within segment.

Returns

NULL if out of range, or corresponding virtual address.

6.6.3.7 static uint64_t metal_io_read (struct metal_io_region * *io,* unsigned long *offset,* memory_order *order,* int *width*) [inline], [static]

Read a value from an I/O region.

Parameters

in	*io*	I/O region handle.
in	*offset*	Offset into I/O region.
in	*order*	Memory ordering.
in	*width*	Width in bytes of datatype to read. This must be 1, 2, 4, or 8, and a compile time constant for this function to inline cleanly.

Returns

Value.

6.6.3.8 static size_t metal_io_region_size (struct metal_io_region * *io*) [inline], [static]

Get size of I/O region.

Parameters

in	*io*	I/O region handle.

Returns

Size of I/O region.

6.6.3.9 static void* metal_io_virt (struct metal_io_region * *io,* unsigned long *offset*) [inline], [static]

Get virtual address for a given offset into the I/O region.

Parameters

in	*io*	I/O region handle.
in	*offset*	Offset into shared memory segment.

Returns

NULL if offset is out of range, or pointer to offset.

6.6.3.10 static unsigned long metal_io_virt_to_offset (struct metal_io_region * *io,* void * *virt*) [inline], [static]

Convert a virtual address to offset within I/O region.

Parameters

in	*io*	I/O region handle.
in	*virt*	Virtual address within segment.

Returns

METAL_BAD_OFFSET if out of range, or offset.

6.6.3.11 static metal_phys_addr_t metal_io_virt_to_phys (struct metal_io_region * *io*, void * *virt*) [inline], [static]

Convert a virtual address to physical address.

Parameters

in	*io*	Shared memory segment handle.
in	*virt*	Virtual address within segment.

Returns

METAL_BAD_PHYS if out of range, or corresponding physical address.

6.6.3.12 static void metal_io_write (struct metal_io_region * *io*, unsigned long *offset*, uint64_t *value*, memory_order *order*, int *width*) [inline], [static]

Write a value into an I/O region.

Parameters

in	*io*	I/O region handle.
in	*offset*	Offset into I/O region.
in	*value*	Value to write.
in	*order*	Memory ordering.
in	*width*	Width in bytes of datatype to read. This must be 1, 2, 4, or 8, and a compile time constant for this function to inline cleanly.

6.6.3.13 void* metal_memcpy_io (void * *dst*, const void * *src*, size_t *size*)

libmetal copy to target memory

This function is to copy specified memory area. The source memory or the destination memory can be device memory.

Parameters

in	*dst*	target memory
in	*src*	source memory
in	*size*	size of memory to copy.

Returns

pointer to the target memory

6.6.3.14 void* metal_memset_io (void * *dst*, int *c*, size_t *size*)

libmetal set device memory

This function is to fill the device memory with the specified value.

Parameters

in	*dst*	target memory
in	*c*	val to fill
in	*size*	size of memory to fill.

Returns

pointer to the target memory

6.6.4 Variable Documentation

6.6.4.1 void(∗ metal_io_ops::close)(struct metal_io_region ∗io)

6.6.4.2 unsigned int metal_io_region::mem_flags

6.6.4.3 struct metal_io_ops metal_io_region::ops

6.6.4.4 metal_phys_addr_t metal_io_region::page_mask

6.6.4.5 unsigned long metal_io_region::page_shift

6.6.4.6 const metal_phys_addr_t∗ metal_io_region::physmap

6.6.4.7 uint64_t(∗ metal_io_ops::read)(struct metal_io_region ∗io, unsigned long offset, memory_order order, int width)

6.6.4.8 size_t metal_io_region::size

6.6.4.9 void∗ metal_io_region::virt

6.6.4.10 void(∗ metal_io_ops::write)(struct metal_io_region ∗io, unsigned long offset, uint64_t value, memory_order order, int width)

6.7 Interrupt Handling Interfaces

Macros

- #define METAL_IRQ_NOT_HANDLED 0
- #define METAL_IRQ_HANDLED 1

Typedefs

- typedef int(∗ metal_irq_handler)(int irq, void ∗priv)

  type of interrupt handler

Functions

- int metal_irq_register (int irq, metal_irq_handler irq_handler, struct metal_device ∗dev, void ∗drv_id)

  register interrupt register interrupt handling of a specific interrupt.
- unsigned int metal_irq_save_disable (void)

  disable interrupts
- void metal_irq_restore_enable (unsigned int flags)

  restore interrupts to their previous state
- void metal_irq_enable (unsigned int vector)

  metal_irq_enable
- void metal_irq_disable (unsigned int vector)

  metal_irq_disable

6.7.1 Detailed Description

6.7.2 Macro Definition Documentation

6.7.2.1 #define METAL_IRQ_HANDLED 1

6.7.2.2 #define METAL_IRQ_NOT_HANDLED 0

IRQ handled status

6.7.3 Typedef Documentation

6.7.3.1 typedef int(∗ metal_irq_handler)(int irq, void ∗priv)

type of interrupt handler

Parameters

in	*interrupt*	id
in	*priv*	private data

Returns

irq handled status

6.7.4 Function Documentation

6.7.4.1 void metal_irq_disable (unsigned int *vector*)

metal_irq_disable

Disables the given interrupt

Parameters

vector	- interrupt vector number

6.7.4.2 void metal_irq_enable (unsigned int *vector*)

metal_irq_enable

Enables the given interrupt

Parameters

vector	- interrupt vector number

6.7.4.3 int metal_irq_register (int *irq,* metal_irq_handler *irq_handler,* struct metal_device * *dev,* void * *drv_id*)

register interrupt register interrupt handling of a specific interrupt.

Parameters

in	*irq*	interrupt id
in	*irq_handler*	interrupt handler If this parameter is NULL, it will deregister the irq handler. If irq_handler, dev, and drv_id are all NULL, it will deregister all the handler of the irq. If irq_handler is NULL, but not both dev and drv_id are NULL, it will only deregister the handler which has been registered with the same dev and drv_id.
in	*dev*	metal device this irq belongs to
in	*drv_id*	driver id. It can be used for driver data.

Returns

0 for success, non-zero on failure

6.7.4.4 void metal_irq_restore_enable (unsigned int *flags*)

restore interrupts to their previous state

Parameters

in	*flags*	previous interrupts state

6.7.4.5 unsigned int metal_irq_save_disable (void)

disable interrupts

Returns

interrupts state

6.8 List Primitives

Data Structures

- struct metal_list

Macros

- #define METAL_DECLARE_LIST(name) struct metal_list name = { .next = &name, .prev = &name }
- #define metal_list_for_each(list, node)

Functions

- static void metal_list_init (struct metal_list *list)
- static void metal_list_add_before (struct metal_list *node, struct metal_list *new_node)
- static void metal_list_add_after (struct metal_list *node, struct metal_list *new_node)
- static void metal_list_add_head (struct metal_list *list, struct metal_list *node)
- static void metal_list_add_tail (struct metal_list *list, struct metal_list *node)
- static int metal_list_is_empty (struct metal_list *list)
- static void metal_list_del (struct metal_list *node)
- static struct metal_list * metal_list_first (struct metal_list *list)

6.8.1 Detailed Description

6.8.2 Macro Definition Documentation

6.8.2.1 #define METAL_DECLARE_LIST(*name*) struct metal_list name = { .next = &name, .prev = &name }

6.8.2.2 #define metal_list_for_each(*list, node*)

Value:

```
for ((node) = (list)->next;             \
             (node) != (list);                  \
             (node) = (node)->next)
```

6.8.3 Function Documentation

6.8.3.1 static void metal_list_add_after (struct metal_list * *node,* struct metal_list * *new_node*) [inline], [static]

6.8.3.2 static void metal_list_add_before (struct metal_list * *node,* struct metal_list * *new_node*) [inline], [static]

6.8.3.3 static void metal_list_add_head (struct metal_list * *list,* struct metal_list * *node*) [inline], [static]

6.8.3.4 static void metal_list_add_tail (struct metal_list * *list,* struct metal_list * *node*) [inline], [static]

6.8.3.5 static void metal_list_del (struct metal_list * *node*) [inline], [static]

6.8.3.6 static struct metal_list* metal_list_first (struct metal_list * *list*) [static]

6.8.3.7 static void metal_list_init (struct metal_list * *list*) [inline], [static]

6.8.3.8 static int metal_list_is_empty (struct metal_list * *list*) [inline],[static]

6.9 Library Logging Interfaces

Macros

- #define metal_log(level,...)

Typedefs

- typedef void(* metal_log_handler)(enum metal_log_level level, const char *format,...)

Enumerations

- enum metal_log_level {
  LOG_EMERGENCY, LOG_ALERT, LOG_CRITICAL, LOG_ERROR,
  LOG_WARNING, LOG_NOTICE, LOG_INFO, LOG_DEBUG }

Functions

- void metal_set_log_handler (metal_log_handler handler)

  Set libmetal log handler.
- metal_log_handler metal_get_log_handler (void)

  Get the current libmetal log handler.
- void metal_set_log_level (enum metal_log_level level)

  Set the level for libmetal logging.
- enum metal_log_level metal_get_log_level (void)

  Get the current level for libmetal logging.
- void metal_default_log_handler (enum metal_log_level level, const char *format,...)

  Default libmetal log handler. This handler prints libmetal log mesages to stderr.

6.9.1 Detailed Description

6.9.2 Macro Definition Documentation

6.9.2.1 #define metal_log(*level, ...*)

Value:

```
(void)((level <= _metal.common.log_level && _metal.
      common.log_handler) \
               ? _metal.common.log_handler(level, __VA_ARGS__)                \
               : 0)
```

Emit a log message if the log level permits.

Parameters

level	Log level.
...	Format string and arguments.

6.9.3 Typedef Documentation

6.9.3.1 typedef void(* metal_log_handler)(enum metal_log_level level, const char *format,...)

Log message handler type.

6.9.4 Enumeration Type Documentation

6.9.4.1 enum metal_log_level

Log message priority levels for libmetal.

Enumerator

LOG_EMERGENCY system is unusable.

LOG_ALERT action must be taken immediately.

LOG_CRITICAL critical conditions.

LOG_ERROR error conditions.

LOG_WARNING warning conditions.

LOG_NOTICE normal but significant condition.

LOG_INFO informational messages.

LOG_DEBUG debug-level messages.

6.9.5 Function Documentation

6.9.5.1 void metal_default_log_handler (enum metal_log_level *level*, const char * *format*, ...)

Default libmetal log handler. This handler prints libmetal log mesages to stderr.

Parameters

in	*level*	log message level.
in	*format*	log message format string.

Returns

0 on success, or -errno on failure.

6.9.5.2 metal_log_handler metal_get_log_handler (void)

Get the current libmetal log handler.

Returns

Current log handler.

6.9.5.3 enum metal_log_level metal_get_log_level (void)

Get the current level for libmetal logging.

Returns

Current log level.

6.9.5.4 void metal_set_log_handler (metal_log_handler *handler*)

Set libmetal log handler.

Parameters

in	*handler*	log message handler.

Returns

0 on success, or -errno on failure.

6.9.5.5 void metal_set_log_level (enum metal_log_level *level*)

Set the level for libmetal logging.

Parameters

in	*level*	log message level.

6.10 Mutex Interfaces

Functions

- static void metal_mutex_init (metal_mutex_t *mutex)

  Initialize a libmetal mutex.
- static void metal_mutex_deinit (metal_mutex_t *mutex)

  Deinitialize a libmetal mutex.
- static int metal_mutex_try_acquire (metal_mutex_t *mutex)

  Try to acquire a mutex.
- static void metal_mutex_acquire (metal_mutex_t *mutex)

  Acquire a mutex.
- static void metal_mutex_release (metal_mutex_t *mutex)

  Release a previously acquired mutex.
- static int metal_mutex_is_acquired (metal_mutex_t *mutex)

  Checked if a mutex has been acquired.

6.10.1 Detailed Description

6.10.2 Function Documentation

6.10.2.1 static void metal_mutex_acquire (metal_mutex_t * *mutex*) [inline], [static]

Acquire a mutex.

Parameters

in	*mutex*	Mutex to mutex.

6.10.2.2 static void metal_mutex_deinit (metal_mutex_t * *mutex*) [inline], [static]

Deinitialize a libmetal mutex.

Parameters

in	*mutex*	Mutex to deinitialize.

6.10.2.3 static void metal_mutex_init (metal_mutex_t * *mutex*) [inline], [static]

Initialize a libmetal mutex.

Parameters

in	*mutex*	Mutex to initialize.

6.10.2.4 static int metal_mutex_is_acquired (metal_mutex_t * *mutex*) [inline], [static]

Checked if a mutex has been acquired.

Parameters

in	*mutex*	mutex to check.

See Also

metal_mutex_try_acquire, metal_mutex_acquire

6.10.2.5 static void metal_mutex_release (metal_mutex_t * *mutex*) [inline], [static]

Release a previously acquired mutex.

Parameters

in	*mutex*	Mutex to mutex.

See Also

metal_mutex_try_acquire, metal_mutex_acquire

6.10.2.6 static int metal_mutex_try_acquire (metal_mutex_t * *mutex*) [inline], [static]

Try to acquire a mutex.

Parameters

in	*mutex*	Mutex to mutex.

Returns

0 on failure to acquire, non-zero on success.

6.11 Shared Memory Interfaces

Data Structures

- struct metal_generic_shmem

Functions

- int metal_shmem_open (const char *name, size_t size, struct metal_io_region **io)

  Open a libmetal shared memory segment.
- int metal_shmem_register_generic (struct metal_generic_shmem *shmem)

  Statically register a generic shared memory region.

6.11.1 Detailed Description

6.11.2 Function Documentation

6.11.2.1 int metal_shmem_open (const char * *name*, size_t *size*, struct metal_io_region ** *io*)

Open a libmetal shared memory segment.

Open a shared memory segment.

Parameters

in	*name*	Name of segment to open.
in	*size*	Size of segment.
out	*io*	I/O region handle, if successful.

Returns

0 on success, or -errno on failure.

See Also

metal_shmem_create

6.11.2.2 int metal_shmem_register_generic (struct metal_generic_shmem * *shmem*)

Statically register a generic shared memory region.

Shared memory regions may be statically registered at application initialization, or may be dynamically opened. This interface is used for static registration of regions. Subsequent calls to metal_shmem_open() look up in this list of pre-registered regions.

Parameters

in	*shmem*	Generic shmem structure.

Returns

0 on success, or -errno on failure.

6.12 Sleep Interfaces

Functions

- int metal_sleep_usec (unsigned int usec)

  delay in microseconds delay the next execution in the calling thread fo usec microseconds.

6.12.1 Detailed Description

6.12.2 Function Documentation

6.12.2.1 int metal_sleep_usec (unsigned int *usec*)

delay in microseconds delay the next execution in the calling thread fo usec microseconds.

Parameters

in	*usec*	microsecond intervals

Returns

0 on success, non-zero for failures

6.13 Spinlock Interfaces

Data Structures

- struct metal_spinlock

Macros

- #define METAL_SPINLOCK_INIT {ATOMIC_VAR_INIT(0)}

Functions

- static void metal_spinlock_init (struct metal_spinlock *slock)

  Initialize a libmetal spinlock.
- static void metal_spinlock_acquire (struct metal_spinlock *slock)

  Acquire a spinlock.
- static void metal_spinlock_release (struct metal_spinlock *slock)

  Release a previously acquired spinlock.

6.13.1 Detailed Description

6.13.2 Macro Definition Documentation

6.13.2.1 #define METAL_SPINLOCK_INIT {ATOMIC_VAR_INIT(0)}

Static metal spinlock initialization.

6.13.3 Function Documentation

6.13.3.1 static void metal_spinlock_acquire (struct metal_spinlock * *slock*) [inline], [static]

Acquire a spinlock.

Parameters

in	*slock*	Spinlock to acquire.

See Also

metal_spinlock_release

6.13.3.2 static void metal_spinlock_init (struct metal_spinlock * *slock*) [inline], [static]

Initialize a libmetal spinlock.

Parameters

in	*slock*	Spinlock to initialize.

6.13.3.3 static void metal_spinlock_release (struct metal_spinlock * *slock*) [inline], [static]

Release a previously acquired spinlock.

Parameters

in	*slock*	Spinlock to release.

See Also

metal_spinlock_acquire

6.14 Top Level Interfaces

Data Structures

- struct metal_init_params
- struct metal_common_state

Macros

- #define METAL_BAD_OFFSET ((unsigned long)-1)
- #define METAL_BAD_PHYS ((metal_phys_addr_t)-1)
- #define METAL_BAD_IRQ ((metal_irq_t)-1)
- #define METAL_INIT_DEFAULTS

Typedefs

- typedef unsigned long metal_phys_addr_t
- typedef int metal_irq_t

Functions

- int metal_init (const struct metal_init_params *params)

  Initialize libmetal.
- void metal_finish (void)

  Shutdown libmetal.

Variables

- struct metal_state _metal

6.14.1 Detailed Description

6.14.2 Macro Definition Documentation

6.14.2.1 #define METAL_BAD_IRQ ((metal_irq_t)-1)

Bad IRQ.

6.14.2.2 #define METAL_BAD_OFFSET ((unsigned long)-1)

Bad offset into shared memory or I/O region.

6.14.2.3 #define METAL_BAD_PHYS ((metal_phys_addr_t)-1)

Bad physical address value.

6.14.2.4 #define METAL_INIT_DEFAULTS

Value:

```
{                                                   \
        .log_handler    = metal_default_log_handler,    \
        .log_level      = LOG_INFO,                     \
}
```

6.14.3 Typedef Documentation

6.14.3.1 typedef int metal_irq_t

Interrupt request number.

6.14.3.2 typedef unsigned long metal_phys_addr_t

Physical address type.

6.14.4 Function Documentation

6.14.4.1 void metal_finish (void)

Shutdown libmetal.

Shutdown the libmetal library, and release all reserved resources.

See Also

metal_init

6.14.4.2 int metal_init (const struct metal_init_params * *params*)

Initialize libmetal.

Initialize the libmetal library.

Parameters

in	*params*	Initialization params (

See Also

metal_init_params).

Returns

0 on success, or -errno on failure.

See Also

metal_finish

6.14.5 Variable Documentation

6.14.5.1 struct metal_state _metal

System specific runtime data.

6.15 TIME Interfaces

Functions

- unsigned long long metal_get_timestamp (void)

  get timestamp This function returns the timestampe as unsigned long long value.

6.15.1 Detailed Description

6.15.2 Function Documentation

6.15.2.1 unsigned long long metal_get_timestamp (void)

get timestamp This function returns the timestampe as unsigned long long value.

Returns

timestamp

6.16 Simple Utilities

Macros

- #define metal_unused(x) do { (x) = (x); } while (0)
- #define metal_dim(x) (sizeof(x) / sizeof(x[0]))
- #define metal_min(x, y) ((x) < (y) ? (x) : (y))
- #define metal_max(x, y) ((x) > (y) ? (x) : (y))
- #define metal_sign(x) ((x) < 0 ? -1 : ((x) > 0 ? 1 : 0))
- #define metal_align_down(size, align) ((size) & ~((align) - 1))
- #define metal_align_up(size, align) metal_align_down((size) + (align) - 1, align)
- #define metal_div_round_down(num, den) ((num) / (den))
- #define metal_div_round_up(num, den) metal_div_round_down((num) + (den) - 1, (den))
- #define metal_ptr_align_down(ptr, align) (void *)(metal_align_down((uintptr_t)(ptr), (uintptr_t)(align)))
- #define metal_ptr_align_up(ptr, align) (void *)(metal_align_up((uintptr_t)(ptr), (uintptr_t)(align)))
- #define metal_offset_of(structure, member) ((uintptr_t) &(((structure *) 0)->member))
- #define metal_container_of(ptr, structure, member) (void *)((uintptr_t)(ptr) - metal_offset_of(structure, member))
- #define METAL_BITS_PER_ULONG (8 * sizeof(unsigned long))
- #define metal_bit(bit) (1UL << (bit))
- #define metal_bitmap_longs(x) metal_div_round_up((x), METAL_BITS_PER_ULONG)
- #define metal_bitmap_for_each_set_bit(bitmap, bit, max)
- #define metal_bitmap_for_each_clear_bit(bitmap, bit, max)

Functions

- static void metal_bitmap_set_bit (unsigned long *bitmap, int bit)
- static int metal_bitmap_is_bit_set (unsigned long *bitmap, int bit)
- static void metal_bitmap_clear_bit (unsigned long *bitmap, int bit)
- static int metal_bitmap_is_bit_clear (unsigned long *bitmap, int bit)
- static unsigned int metal_bitmap_next_set_bit (unsigned long *bitmap, unsigned int start, unsigned int max)
- static unsigned int metal_bitmap_next_clear_bit (unsigned long *bitmap, unsigned int start, unsigned int max)
- static unsigned long metal_log2 (unsigned long in)

6.16.1 Detailed Description

6.16.2 Macro Definition Documentation

6.16.2.1 #define metal_align_down(*size, align*) ((size) & ~((align) - 1))

Align 'size' down to a multiple of 'align' (must be a power of two).

6.16.2.2 #define metal_align_up(*size, align*) metal_align_down((size) + (align) - 1, align)

Align 'size' up to a multiple of 'align' (must be a power of two).

6.16.2.3 #define metal_bit(*bit*) (1UL << (bit))

6.16.2.4 #define metal_bitmap_for_each_clear_bit(*bitmap, bit, max*)

Value:

```
for ((bit) = metal_bitmap_next_clear_bit((bitmap), 0, (max));        \
             (bit) < (max);                                          \
             (bit) = metal_bitmap_next_clear_bit((bitmap), (bit), (max)))
```

6.16.2.5 #define metal_bitmap_for_each_set_bit(*bitmap, bit, max*)

Value:

```
for ((bit) = metal_bitmap_next_set_bit((bitmap), 0, (max));     \
             (bit) < (max);                                      \
             (bit) = metal_bitmap_next_set_bit((bitmap), (bit), (max)))
```

6.16.2.6 #define metal_bitmap_longs(*x*) metal_div_round_up((x), METAL_BITS_PER_ULONG)

6.16.2.7 #define METAL_BITS_PER_ULONG (8 ∗ sizeof(unsigned long))

6.16.2.8 #define metal_container_of(*ptr, structure, member*) (void ∗)((uintptr_t)(ptr) - metal_offset_of(structure, member))

Compute pointer to a structure given a pointer to one of its fields.

6.16.2.9 #define metal_dim(*x*) (sizeof(x) / sizeof(x[0]))

Figure out number of elements in an array.

6.16.2.10 #define metal_div_round_down(*num, den*) ((num) / (den))

Divide (and round down).

6.16.2.11 #define metal_div_round_up(*num, den*) metal_div_round_down((num) + (den) - 1, (den))

Divide (and round up).

6.16.2.12 #define metal_max(*x, y*) ((x) > (y) ? (x) : (y))

Maximum of two numbers (warning: multiple evaluation!).

6.16.2.13 #define metal_min(*x, y*) ((x) < (y) ? (x) : (y))

Minimum of two numbers (warning: multiple evaluation!).

6.16.2.14 #define metal_offset_of(*structure, member*) ((uintptr_t) &(((structure ∗) 0)->member))

Compute offset of a field within a structure.

6.16.2.15 #define metal_ptr_align_down(*ptr, align*) (void ∗)(metal_align_down((uintptr_t)(ptr), (uintptr_t)(align)))

Align 'ptr' down to a multiple of 'align' (must be a power of two).

6.16.2.16 #define metal_ptr_align_up(*ptr, align*) (void ∗)(metal_align_up((uintptr_t)(ptr), (uintptr_t)(align)))

Align 'ptr' up to a multiple of 'align' (must be a power of two).

6.16.2.17 #define metal_sign(x) ((x) < 0 ? -1 : ((x) > 0 ? 1 : 0))

Sign of a number [-1, 0, or 1] (warning: multiple evaluation!).

6.16.2.18 #define metal_unused(x) do { (x) = (x); } while (0)

Marker for unused function arguments/variables.

6.16.3 Function Documentation

6.16.3.1 static void metal_bitmap_clear_bit (unsigned long * bitmap, int bit) [inline], [static]

6.16.3.2 static int metal_bitmap_is_bit_clear (unsigned long * bitmap, int bit) [inline], [static]

6.16.3.3 static int metal_bitmap_is_bit_set (unsigned long * bitmap, int bit) [inline], [static]

6.16.3.4 static unsigned int metal_bitmap_next_clear_bit (unsigned long * bitmap, unsigned int start, unsigned int max) [inline], [static]

6.16.3.5 static unsigned int metal_bitmap_next_set_bit (unsigned long * bitmap, unsigned int start, unsigned int max) [inline], [static]

6.16.3.6 static void metal_bitmap_set_bit (unsigned long * bitmap, int bit) [inline], [static]

6.16.3.7 static unsigned long metal_log2 (unsigned long in) [inline], [static]

6.17 Library Version Interfaces

Functions

- int metal_ver_major (void)

  Library major version number.
- int metal_ver_minor (void)

  Library minor version number.
- int metal_ver_patch (void)

  Library patch level.
- const char * metal_ver (void)

  Library version string.

6.17.1 Detailed Description

6.17.2 Function Documentation

6.17.2.1 const char* metal_ver (void)

Library version string.

Return the version string of the library linked into the application. This could differ from the value of METAL_VER, which is the version string of the library that the application was compiled against.

Returns

Library version string.

See Also

METAL_VER

6.17.2.2 int metal_ver_major (void)

Library major version number.

Return the major version number of the library linked into the application. This is required to match the value of METAL_VER_MAJOR, which is the major version of the library that the application was compiled against.

Returns

Library major version number.

See Also

METAL_VER_MAJOR

6.17.2.3 int metal_ver_minor (void)

Library minor version number.

Return the minor version number of the library linked into the application. This could differ from the value of METAL_VER_MINOR, which is the minor version of the library that the application was compiled against.

Returns

Library minor version number.

See Also

METAL_VER_MINOR

6.17.2.4 int metal_ver_patch (void)

Library patch level.

Return the patch level of the library linked into the application. This could differ from the value of METAL_VER_PATCH, which is the patch level of the library that the application was compiled against.

Returns

Library patch level.

See Also

METAL_VER_PATCH

Chapter 7

Data Structure Documentation

7.1 linux_bus Struct Reference

Data Fields

- struct metal_bus bus
- const char * bus_name
- struct linux_driver drivers [MAX_DRIVERS]
- struct sysfs_bus * sbus

7.1.1 Field Documentation

7.1.1.1 struct metal_bus linux_bus::bus

7.1.1.2 const char* linux_bus::bus_name

7.1.1.3 struct linux_driver linux_bus::drivers[MAX_DRIVERS]

7.1.1.4 struct sysfs_bus* linux_bus::sbus

The documentation for this struct was generated from the following file:

- lib/system/linux/device.c

7.2 linux_device Struct Reference

Data Fields

- struct metal_device device
- char dev_name [PATH_MAX]
- char dev_path [PATH_MAX]
- char cls_path [PATH_MAX]
- metal_phys_addr_t region_phys [METAL_MAX_DEVICE_REGIONS]
- struct linux_driver * ldrv
- struct sysfs_device * sdev
- struct sysfs_attribute * override
- int fd

7.2.1 Field Documentation

7.2.1.1 char linux_device::cls_path[PATH_MAX]

7.2.1.2 char linux_device::dev_name[PATH_MAX]

7.2.1.3 char linux_device::dev_path[PATH_MAX]

7.2.1.4 struct metal_device linux_device::device

7.2.1.5 int linux_device::fd

7.2.1.6 struct linux_driver* linux_device::ldrv

7.2.1.7 struct sysfs_attribute* linux_device::override

7.2.1.8 metal_phys_addr_t linux_device::region_phys[METAL_MAX_DEVICE_REGIONS]

7.2.1.9 struct sysfs_device* linux_device::sdev

The documentation for this struct was generated from the following file:

- lib/system/linux/device.c

7.3 linux_driver Struct Reference

Data Fields

- const char * drv_name
- const char * mod_name
- const char * cls_name
- struct sysfs_driver * sdrv
- int(* dev_open)(struct linux_bus *lbus, struct linux_device *ldev)
- void(* dev_close)(struct linux_bus *lbus, struct linux_device *ldev)
- void(* dev_irq_ack)(struct linux_bus *lbus, struct linux_device *ldev, int irq)
- int(* dev_dma_map)(struct linux_bus *lbus, struct linux_device *ldev, uint32_t dir, struct metal_sg *sg_in, int nents_in, struct metal_sg *sg_out)
- void(* dev_dma_unmap)(struct linux_bus *lbus, struct linux_device *ldev, uint32_t dir, struct metal_sg *sg, int nents)

7.3.1 Field Documentation

7.3.1.1 const char* linux_driver::cls_name

7.3.1.2 void(* linux_driver::dev_close)(struct linux_bus *lbus, struct linux_device *ldev)

7.3.1.3 int(* linux_driver::dev_dma_map)(struct linux_bus *lbus, struct linux_device *ldev, uint32_t dir, struct metal_sg *sg_in, int nents_in, struct metal_sg *sg_out)

7.3.1.4 void(* linux_driver::dev_dma_unmap)(struct linux_bus *lbus, struct linux_device *ldev, uint32_t dir, struct metal_sg *sg, int nents)

7.3.1.5 void(* linux_driver::dev_irq_ack)(struct linux_bus *lbus, struct linux_device *ldev, int irq)

7.3.1.6 int(* linux_driver::dev_open)(struct linux_bus *lbus, struct linux_device *ldev)

7.3.1.7 const char* linux_driver::drv_name

7.3.1.8 const char* linux_driver::mod_name

7.3.1.9 struct sysfs_driver* linux_driver::sdrv

The documentation for this struct was generated from the following file:

- lib/system/linux/device.c

7.4 metal_bus Struct Reference

`#include <device.h>`

Data Fields

- const char * name
- struct metal_bus_ops ops
- struct metal_list devices
- struct metal_list node

7.4.1 Detailed Description

Libmetal bus structure.

7.4.2 Field Documentation

7.4.2.1 struct metal_list metal_bus::devices

7.4.2.2 const char* metal_bus::name

7.4.2.3 struct metal_list metal_bus::node

7.4.2.4 struct metal_bus_ops metal_bus::ops

The documentation for this struct was generated from the following file:

- lib/device.h

7.5 metal_bus_ops Struct Reference

`#include <device.h>`

Data Fields

- void(* bus_close)(struct metal_bus *bus)
- int(* dev_open)(struct metal_bus *bus, const char *dev_name, struct metal_device **device)
- void(* dev_close)(struct metal_bus *bus, struct metal_device *device)

- void(∗ dev_irq_ack)(struct metal_bus ∗bus, struct metal_device ∗device, int irq)
- int(∗ dev_dma_map)(struct metal_bus ∗bus, struct metal_device ∗device, uint32_t dir, struct metal_sg ∗sg_in, int nents_in, struct metal_sg ∗sg_out)
- void(∗ dev_dma_unmap)(struct metal_bus ∗bus, struct metal_device ∗device, uint32_t dir, struct metal_sg ∗sg, int nents)

7.5.1 Detailed Description

Bus operations.

7.5.2 Field Documentation

7.5.2.1 void(∗ metal_bus_ops::bus_close)(struct metal_bus ∗bus)

7.5.2.2 void(∗ metal_bus_ops::dev_close)(struct metal_bus ∗bus, struct metal_device ∗device)

7.5.2.3 int(∗ metal_bus_ops::dev_dma_map)(struct metal_bus ∗bus, struct metal_device ∗device, uint32_t dir, struct metal_sg ∗sg_in, int nents_in, struct metal_sg ∗sg_out)

7.5.2.4 void(∗ metal_bus_ops::dev_dma_unmap)(struct metal_bus ∗bus, struct metal_device ∗device, uint32_t dir, struct metal_sg ∗sg, int nents)

7.5.2.5 void(∗ metal_bus_ops::dev_irq_ack)(struct metal_bus ∗bus, struct metal_device ∗device, int irq)

7.5.2.6 int(∗ metal_bus_ops::dev_open)(struct metal_bus ∗bus, const char ∗dev_name, struct metal_device ∗∗device)

The documentation for this struct was generated from the following file:

- lib/device.h

7.6 metal_common_state Struct Reference

`#include <sys.h>`

Data Fields

- enum metal_log_level log_level
- metal_log_handler log_handler
- struct metal_list bus_list
- struct metal_list generic_shmem_list
- struct metal_list generic_device_list

7.6.1 Detailed Description

System independent runtime state for libmetal. This is part of a system specific singleton data structure (

See Also

_metal).

7.6.2 Field Documentation

7.6.2.1 struct metal_list metal_common_state::bus_list

List of registered buses.

7.6.2.2 struct metal_list metal_common_state::generic_device_list

Generic statically defined devices.

7.6.2.3 struct metal_list metal_common_state::generic_shmem_list

Generic statically defined shared memory segments.

7.6.2.4 metal_log_handler metal_common_state::log_handler

Current log handler (null for none).

7.6.2.5 enum metal_log_level metal_common_state::log_level

Current log level.

The documentation for this struct was generated from the following file:

- lib/sys.h

7.7 metal_condition Struct Reference

`#include <condition.h>`

Data Fields

- metal_mutex_t * m
- atomic_int v
- atomic_int waiters
- atomic_int wakeups

7.7.1 Field Documentation

7.7.1.1 metal_mutex_t * metal_condition::m

mutex. The condition variable is attached to this mutex when it is waiting. It is also used to check correctness in case there are multiple waiters.

7.7.1.2 atomic_int metal_condition::v

condition variable value.

7.7.1.3 atomic_int metal_condition::waiters

number of waiters.

7.7.1.4 atomic_int metal_condition::wakeups

number of wakeups.

The documentation for this struct was generated from the following file:

- lib/system/freertos/condition.h

7.8 metal_device Struct Reference

`#include <device.h>`

Data Fields

- const char * name
- struct metal_bus * bus
- unsigned num_regions
- struct metal_io_region regions [METAL_MAX_DEVICE_REGIONS]
- struct metal_list node
- int irq_num
- void * irq_info

7.8.1 Detailed Description

Libmetal device structure.

7.8.2 Field Documentation

7.8.2.1 struct metal_bus* metal_device::bus

7.8.2.2 void* metal_device::irq_info

7.8.2.3 int metal_device::irq_num

7.8.2.4 const char* metal_device::name

7.8.2.5 struct metal_list metal_device::node

7.8.2.6 unsigned metal_device::num_regions

7.8.2.7 struct metal_io_region metal_device::regions[METAL_MAX_DEVICE_REGIONS]

The documentation for this struct was generated from the following file:

- lib/device.h

7.9 metal_generic_shmem Struct Reference

`#include <shmem.h>`

Data Fields

- const char * name
- struct metal_io_region io
- struct metal_list node

7.9.1 Detailed Description

Generic shared memory data structure.

7.9.2 Field Documentation

7.9.2.1 struct metal_io_region metal_generic_shmem::io

7.9.2.2 const char* metal_generic_shmem::name

7.9.2.3 struct metal_list metal_generic_shmem::node

The documentation for this struct was generated from the following file:

- lib/shmem.h

7.10 metal_init_params Struct Reference

```
#include <sys.h>
```

Data Fields

- metal_log_handler log_handler
- enum metal_log_level log_level

7.10.1 Detailed Description

Initialization configuration for libmetal.

7.10.2 Field Documentation

7.10.2.1 metal_log_handler metal_init_params::log_handler

log message handler (defaults to stderr).

7.10.2.2 enum metal_log_level metal_init_params::log_level

default log message level (defaults to emergency).

The documentation for this struct was generated from the following file:

- lib/sys.h

7.11 metal_io_ops Struct Reference

`#include <io.h>`

Data Fields

- uint64_t(∗ read)(struct metal_io_region ∗io, unsigned long offset, memory_order order, int width)
- void(∗ write)(struct metal_io_region ∗io, unsigned long offset, uint64_t value, memory_order order, int width)
- void(∗ close)(struct metal_io_region ∗io)

7.11.1 Detailed Description

Generic I/O operations.

The documentation for this struct was generated from the following file:

- lib/io.h

7.12 metal_io_region Struct Reference

`#include <io.h>`

Data Fields

- void ∗ virt
- const metal_phys_addr_t ∗ physmap
- size_t size
- unsigned long page_shift
- metal_phys_addr_t page_mask
- unsigned int mem_flags
- struct metal_io_ops ops

7.12.1 Detailed Description

Libmetal I/O region structure.

The documentation for this struct was generated from the following file:

- lib/io.h

7.13 metal_irq_hddesc Struct Reference

Data Fields

- metal_irq_handler hd
- void ∗ drv_id
- struct metal_device ∗ dev

7.13.1 Detailed Description

IRQ handler descriptor structure

7.13.2 Field Documentation

7.13.2.1 struct metal_device* metal_irq_hddesc::dev

metal device

7.13.2.2 void * metal_irq_hddesc::drv_id

id to identify the driver of the irq handler

7.13.2.3 metal_irq_handler metal_irq_hddesc::hd

irq handler

The documentation for this struct was generated from the following file:

- lib/system/freertos/irq.c

7.14 metal_irqs_state Struct Reference

Data Fields

- struct metal_irq_hddesc hds [MAX_IRQS][MAX_HDS]
- unsigned int intr_enable
- metal_mutex_t irq_lock
- signed char irq_reg_stat [MAX_IRQS]
- int irq_reg_fd
- unsigned int irq_state
- pthread_t irq_pthread

7.14.1 Field Documentation

7.14.1.1 struct metal_irq_hddesc metal_irqs_state::hds

irqs handlers descriptors

7.14.1.2 unsigned int metal_irqs_state::intr_enable

7.14.1.3 metal_mutex_t metal_irqs_state::irq_lock

irq handling lock

7.14.1.4 pthread_t metal_irqs_state::irq_pthread

irq handling thread id

7.14.1.5 int metal_irqs_state::irq_reg_fd

irqs registration notification file descriptor

7.14.1.6 signed char metal_irqs_state::irq_reg_stat[MAX_IRQS]

irqs registration statistics. It restore how many handlers have been registered for each IRQ.

7.14.1.7 unsigned int metal_irqs_state::irq_state

global irq handling state

The documentation for this struct was generated from the following file:

- lib/system/freertos/irq.c

7.15 metal_list Struct Reference

`#include <list.h>`

Data Fields

- struct metal_list * next
- struct metal_list * prev

7.15.1 Field Documentation

7.15.1.1 struct metal_list* metal_list::next

7.15.1.2 struct metal_list * metal_list::prev

The documentation for this struct was generated from the following file:

- lib/list.h

7.16 metal_mutex_t Struct Reference

`#include <mutex.h>`

Data Fields

- SemaphoreHandle_t m
- atomic_int v

7.16.1 Field Documentation

7.16.1.1 SemaphoreHandle_t metal_mutex_t::m

7.16.1.2 atomic_int metal_mutex_t::v

The documentation for this struct was generated from the following file:

- lib/system/freertos/mutex.h

7.17 metal_page_size Struct Reference

`#include <sys.h>`

Data Fields

- size_t page_size
- unsigned long page_shift
- char path [PATH_MAX]
- int mmap_flags

7.17.1 Detailed Description

Structure of shared page or hugepage sized data.

7.17.2 Field Documentation

7.17.2.1 int metal_page_size::mmap_flags

Flags to use for mmap.

7.17.2.2 unsigned long metal_page_size::page_shift

Page shift.

7.17.2.3 size_t metal_page_size::page_size

Page size.

7.17.2.4 char metal_page_size::path[PATH_MAX]

Path to hugetlbfs (or tmpfs) mount point.

The documentation for this struct was generated from the following file:

- lib/system/linux/sys.h

7.18 metal_sg Struct Reference

scatter/gather list element structure

`#include <dma.h>`

Data Fields

- void * virt
- struct metal_io_region * io
- int len

7.18.1 Detailed Description

scatter/gather list element structure

7.18.2 Field Documentation

7.18.2.1 struct metal_io_region* metal_sg::io

IO region

7.18.2.2 int metal_sg::len

length

7.18.2.3 void* metal_sg::virt

CPU virtual address

The documentation for this struct was generated from the following file:

- lib/dma.h

7.19 metal_shmem Struct Reference

Data Fields

- struct metal_io_region io
- metal_phys_addr_t * phys

7.19.1 Field Documentation

7.19.1.1 struct metal_io_region metal_shmem::io

7.19.1.2 metal_phys_addr_t* metal_shmem::phys

The documentation for this struct was generated from the following file:

- lib/system/linux/shmem.c

7.20 metal_spinlock Struct Reference

```
#include <spinlock.h>
```

Data Fields

- atomic_int v

7.20.1 Field Documentation

7.20.1.1 atomic_int metal_spinlock::v

The documentation for this struct was generated from the following file:

- lib/spinlock.h

7.21 metal_state Struct Reference

`#include <sys.h>`

Data Fields

- struct metal_common_state common
- int data_fd
- unsigned long page_size
- unsigned long page_shift
- const char * sysfs_path
- const char * tmp_path
- struct metal_page_size page_sizes [MAX_PAGE_SIZES]
- int num_page_sizes
- int pagemap_fd

7.21.1 Detailed Description

Structure for FreeRTOS libmetal runtime state.

Structure of generic libmetal runtime state.

Structure of linux specific libmetal runtime state.

7.21.2 Field Documentation

7.21.2.1 struct metal_common_state metal_state::common

Common (system independent) data.

7.21.2.2 int metal_state::data_fd

file descriptor for shared data.

7.21.2.3 int metal_state::num_page_sizes

number of available page sizes.

7.21.2.4 unsigned long metal_state::page_shift

system page shift.

7.21.2.5 unsigned long metal_state::page_size

system page size.

7.21.2.6 struct metal_page_size metal_state::page_sizes[MAX_PAGE_SIZES]

available page sizes.

7.21.2.7 int metal_state::pagemap_fd

File descriptor for /proc/self/pagemap (or -1).

7.21.2.8 const char* metal_state::sysfs_path

sysfs mount point.

7.21.2.9 const char* metal_state::tmp_path

sysfs mount point.

The documentation for this struct was generated from the following file:

- lib/system/freertos/sys.h

Chapter 8

File Documentation

8.1 lib/alloc.h File Reference

```
#include "metal/system/@PROJECT_SYSTEM@/alloc.h"
```

Functions

- static void * metal_allocate_memory (unsigned int size)

  allocate requested memory size return a pointer to the allocated memory
- static void metal_free_memory (void *ptr)

  free the memory previously allocated

8.2 lib/system/freertos/alloc.h File Reference

```
#include "FreeRTOS.h"
```

Functions

- static void * metal_allocate_memory (unsigned int size)
- static void metal_free_memory (void *ptr)

8.2.1 Function Documentation

8.2.1.1 static void* metal_allocate_memory (unsigned int *size*) [inline], [static]

8.2.1.2 static void metal_free_memory (void * *ptr*) [inline], [static]

8.3 lib/system/generic/alloc.h File Reference

```
#include <stdlib.h>
```

Functions

- static void * metal_allocate_memory (unsigned int size)
- static void metal_free_memory (void *ptr)

8.3.1 Function Documentation

8.3.1.1 static void* metal_allocate_memory (unsigned int *size*) [inline], [static]

8.3.1.2 static void metal_free_memory (void * *ptr*) [inline], [static]

8.4 lib/system/linux/alloc.h File Reference

```
#include <stdlib.h>
```

Functions

- static void * metal_allocate_memory (unsigned int size)
- static void metal_free_memory (void *ptr)

8.4.1 Function Documentation

8.4.1.1 static void* metal_allocate_memory (unsigned int *size*) [inline], [static]

8.4.1.2 static void metal_free_memory (void * *ptr*) [inline], [static]

8.5 lib/atomic.h File Reference

```
#include "metal/config.h"
#include <metal/processor/@PROJECT_PROCESSOR@/atomic.h>
```

8.6 lib/compiler/gcc/atomic.h File Reference

Macros

- #define ATOMIC_FLAG_INIT 0
- #define ATOMIC_VAR_INIT(VAL) (VAL)
- #define atomic_flag_test_and_set(FLAG) __sync_lock_test_and_set((FLAG), 1)
- #define atomic_flag_test_and_set_explicit(FLAG, MO) atomic_flag_test_and_set(FLAG)
- #define atomic_flag_clear(FLAG) __sync_lock_release((FLAG))
- #define atomic_flag_clear_explicit(FLAG, MO) atomic_flag_clear(FLAG)
- #define atomic_init(OBJ, VAL) do { *(OBJ) = (VAL); } while (0)
- #define atomic_is_lock_free(OBJ) (sizeof(*(OBJ)) <= sizeof(long))
- #define atomic_store(OBJ, VAL) do { *(OBJ) = (VAL); __sync_synchronize(); } while (0)
- #define atomic_store_explicit(OBJ, VAL, MO) atomic_store((OBJ), (VAL))
- #define atomic_load(OBJ) ({ __sync_synchronize(); *(OBJ); })
- #define atomic_load_explicit(OBJ, MO) atomic_load(OBJ)
- #define atomic_exchange(OBJ, DES)
- #define atomic_exchange_explicit(OBJ, DES, MO) atomic_exchange((OBJ), (DES))

- #define atomic_compare_exchange_strong(OBJ, EXP, DES)
- #define atomic_compare_exchange_strong_explicit(OBJ, EXP, DES, MO) atomic_compare_exchange_strong((OBJ), (EXP), (DES))
- #define atomic_compare_exchange_weak(OBJ, EXP, DES) atomic_compare_exchange_strong((OBJ), (EXP), (DES))
- #define atomic_compare_exchange_weak_explicit(OBJ, EXP, DES, MO) atomic_compare_exchange_weak((OBJ), (EXP), (DES))
- #define atomic_fetch_add(OBJ, VAL) __sync_fetch_and_add((OBJ), (VAL))
- #define atomic_fetch_add_explicit(OBJ, VAL, MO) atomic_fetch_add((OBJ), (VAL))
- #define atomic_fetch_sub(OBJ, VAL) __sync_fetch_and_sub((OBJ), (VAL))
- #define atomic_fetch_sub_explicit(OBJ, VAL, MO) atomic_fetch_sub((OBJ), (VAL))
- #define atomic_fetch_or(OBJ, VAL) __sync_fetch_and_or((OBJ), (VAL))
- #define atomic_fetch_or_explicit(OBJ, VAL, MO) atomic_fetch_or((OBJ), (VAL))
- #define atomic_fetch_xor(OBJ, VAL) __sync_fetch_and_xor((OBJ), (VAL))
- #define atomic_fetch_xor_explicit(OBJ, VAL, MO) atomic_fetch_xor((OBJ), (VAL))
- #define atomic_fetch_and(OBJ, VAL) __sync_fetch_and_and((OBJ), (VAL))
- #define atomic_fetch_and_explicit(OBJ, VAL, MO) atomic_fetch_and((OBJ), (VAL))
- #define atomic_thread_fence(MO) __sync_synchronize()
- #define atomic_signal_fence(MO) __sync_synchronize()

Typedefs

- typedef int atomic_flag
- typedef char atomic_char
- typedef unsigned char atomic_uchar
- typedef short atomic_short
- typedef unsigned short atomic_ushort
- typedef int atomic_int
- typedef unsigned int atomic_uint
- typedef long atomic_long
- typedef unsigned long atomic_ulong
- typedef long long atomic_llong
- typedef unsigned long long atomic_ullong

Enumerations

- enum memory_order {
  memory_order_relaxed, memory_order_consume, memory_order_acquire, memory_order_release,
  memory_order_acq_rel, memory_order_seq_cst }

8.6.1 Macro Definition Documentation

8.6.1.1 #define atomic_compare_exchange_strong(*OBJ, EXP, DES*)

Value:

```
({                                                              \
                typeof(OBJ) obj = (OBJ);                        \
                typeof(EXP) exp = (EXP);                        \
                typeof(*obj) expval = *exp;                     \
                typeof(*obj) oldval = __sync_val_compare_and_swap(      \
                        obj, expval, (DES));                    \
                *exp = oldval;                                  \
                oldval == expval;                               \
        })
```

8.6.1.2 #define atomic_compare_exchange_strong_explicit(*OBJ, EXP, DES, MO*) atomic_compare_exchange_strong((OBJ), (EXP), (DES))

8.6.1.3 #define atomic_compare_exchange_weak(*OBJ, EXP, DES*) atomic_compare_exchange_strong((OBJ), (EXP), (DES))

8.6.1.4 #define atomic_compare_exchange_weak_explicit(*OBJ, EXP, DES, MO*) atomic_compare_exchange_weak((OBJ), (EXP), (DES))

8.6.1.5 #define atomic_exchange(*OBJ, DES*)

Value:

```
({                                                              \
                typeof(OBJ) obj = (OBJ);                        \
                typeof(*obj) des = (DES);                       \
                typeof(*obj) expval;                            \
                typeof(*obj) oldval = atomic_load(obj);             \
                do {                                            \
                        expval = oldval;                        \
                        oldval = __sync_val_compare_and_swap(   \
                                obj, expval, des);              \
                } while (oldval != expval);                     \
                oldval;                                         \
        })
```

8.6.1.6 #define atomic_exchange_explicit(*OBJ, DES, MO*) atomic_exchange((OBJ), (DES))

8.6.1.7 #define atomic_fetch_add(*OBJ, VAL*) __sync_fetch_and_add((OBJ), (VAL))

8.6.1.8 #define atomic_fetch_add_explicit(*OBJ, VAL, MO*) atomic_fetch_add((OBJ), (VAL))

8.6.1.9 #define atomic_fetch_and(*OBJ, VAL*) __sync_fetch_and_and((OBJ), (VAL))

8.6.1.10 #define atomic_fetch_and_explicit(*OBJ, VAL, MO*) atomic_fetch_and((OBJ), (VAL))

8.6.1.11 #define atomic_fetch_or(*OBJ, VAL*) __sync_fetch_and_or((OBJ), (VAL))

8.6.1.12 #define atomic_fetch_or_explicit(*OBJ, VAL, MO*) atomic_fetch_or((OBJ), (VAL))

8.6.1.13 #define atomic_fetch_sub(*OBJ, VAL*) __sync_fetch_and_sub((OBJ), (VAL))

8.6.1.14 #define atomic_fetch_sub_explicit(*OBJ, VAL, MO*) atomic_fetch_sub((OBJ), (VAL))

8.6.1.15 #define atomic_fetch_xor(*OBJ, VAL*) __sync_fetch_and_xor((OBJ), (VAL))

8.6.1.16 #define atomic_fetch_xor_explicit(*OBJ, VAL, MO*) atomic_fetch_xor((OBJ), (VAL))

8.6.1.17 #define atomic_flag_clear(*FLAG*) __sync_lock_release((FLAG))

8.6.1.18 #define atomic_flag_clear_explicit(*FLAG, MO*) atomic_flag_clear(FLAG)

8.6.1.19 #define ATOMIC_FLAG_INIT 0

8.6.1.20 #define atomic_flag_test_and_set(*FLAG*) __sync_lock_test_and_set((FLAG), 1)

8.6.1.21 #define atomic_flag_test_and_set_explicit(*FLAG, MO*) atomic_flag_test_and_set(FLAG)

8.6.1.22 #define atomic_init(*OBJ, VAL*) do { *(OBJ) = (VAL); } while (0)

8.6.1.23 #define atomic_is_lock_free(*OBJ*) (sizeof(∗(OBJ)) <= sizeof(long))

8.6.1.24 #define atomic_load(*OBJ*) ({ __sync_synchronize(); ∗(OBJ); })

8.6.1.25 #define atomic_load_explicit(*OBJ, MO*) atomic_load(OBJ)

8.6.1.26 #define atomic_signal_fence(*MO*) __sync_synchronize()

8.6.1.27 #define atomic_store(*OBJ, VAL*) do { ∗(OBJ) = (VAL); __sync_synchronize(); } while (0)

8.6.1.28 #define atomic_store_explicit(*OBJ, VAL, MO*) atomic_store((OBJ), (VAL))

8.6.1.29 #define atomic_thread_fence(*MO*) __sync_synchronize()

8.6.1.30 #define ATOMIC_VAR_INIT(*VAL*) (VAL)

8.6.2 Typedef Documentation

8.6.2.1 typedef char atomic_char

8.6.2.2 typedef int atomic_flag

8.6.2.3 typedef int atomic_int

8.6.2.4 typedef long long atomic_llong

8.6.2.5 typedef long atomic_long

8.6.2.6 typedef short atomic_short

8.6.2.7 typedef unsigned char atomic_uchar

8.6.2.8 typedef unsigned int atomic_uint

8.6.2.9 typedef unsigned long long atomic_ullong

8.6.2.10 typedef unsigned long atomic_ulong

8.6.2.11 typedef unsigned short atomic_ushort

8.6.3 Enumeration Type Documentation

8.6.3.1 enum memory_order

Enumerator

memory_order_relaxed
memory_order_consume
memory_order_acquire
memory_order_release
memory_order_acq_rel
memory_order_seq_cst

8.7 lib/processor/aarch64/atomic.h File Reference

8.8 lib/processor/arm/atomic.h File Reference

8.9 lib/processor/x86_64/atomic.h File Reference

8.10 lib/cache.h File Reference

Functions

- void metal_cache_flush (void *addr, unsigned int len)

  flush specified data cache
- void metal_cache_invalidate (void *addr, unsigned int len)

  invalidate specified data cache

8.11 lib/compiler/gcc/compiler.h File Reference

Macros

- #define metal_align(n) __attribute__((aligned(n)))

8.11.1 Macro Definition Documentation

8.11.1.1 #define metal_align(*n*) __attribute__((aligned(n)))

8.12 lib/compiler.h File Reference

8.13 lib/condition.h File Reference

```
#include "metal/mutex.h"
#include "metal/utilities.h"
#include "metal/system/@PROJECT_SYSTEM@/condition.h"
```

Functions

- static void metal_condition_init (struct metal_condition *cv)

  Initialize a libmetal condition variable.
- static int metal_condition_signal (struct metal_condition *cv)

  Notify one waiter. Before calling this function, the caller should have acquired the mutex.
- static int metal_condition_broadcast (struct metal_condition *cv)

  Notify all waiters. Before calling this function, the caller should have acquired the mutex.
- int metal_condition_wait (struct metal_condition *cv, metal_mutex_t *m)

  Block until the condition variable is notified. Before calling this function, the caller should have acquired the mutex.

8.14 lib/system/freertos/condition.h File Reference

```
#include "metal/atomic.h"
```

Data Structures

- struct metal_condition

Macros

- #define METAL_CONDITION_INIT { NULL, ATOMIC_VAR_INIT(0) }

Functions

- static void metal_condition_init (struct metal_condition *cv)
- static int metal_condition_signal (struct metal_condition *cv)
- static int metal_condition_broadcast (struct metal_condition *cv)

8.14.1 Macro Definition Documentation

8.14.1.1 #define METAL_CONDITION_INIT { NULL, ATOMIC_VAR_INIT(0) }

Static metal condition variable initialization.

8.14.2 Function Documentation

8.14.2.1 static int metal_condition_broadcast (struct metal_condition * *cv*) [inline], [static]

8.14.2.2 static void metal_condition_init (struct metal_condition * *cv*) [inline], [static]

8.14.2.3 static int metal_condition_signal (struct metal_condition * *cv*) [inline], [static]

8.15 lib/system/generic/condition.h File Reference

```
#include <unistd.h>
#include "metal/atomic.h"
#include <stdint.h>
#include <limits.h>
#include <errno.h>
```

Data Structures

- struct metal_condition

Macros

- #define METAL_CONDITION_INIT { NULL, ATOMIC_VAR_INIT(0) }

Functions

- static void metal_condition_init (struct metal_condition *cv)
- static int metal_condition_signal (struct metal_condition *cv)
- static int metal_condition_broadcast (struct metal_condition *cv)

8.15.1 Macro Definition Documentation

8.15.1.1 #define METAL_CONDITION_INIT { NULL, ATOMIC_VAR_INIT(0) }

Static metal condition variable initialization.

8.15.2 Function Documentation

8.15.2.1 static int metal_condition_broadcast (struct metal_condition * *cv*) [inline], [static]

8.15.2.2 static void metal_condition_init (struct metal_condition * *cv*) [inline], [static]

8.15.2.3 static int metal_condition_signal (struct metal_condition * *cv*) [inline], [static]

wake up waiters if there are any.

8.16 lib/system/linux/condition.h File Reference

```
#include <unistd.h>
#include <sys/syscall.h>
#include <linux/futex.h>
#include "metal/atomic.h"
#include <stdint.h>
#include <limits.h>
#include <errno.h>
```

Data Structures

- struct metal_condition

Macros

- #define METAL_CONDITION_INIT { NULL, ATOMIC_VAR_INIT(0), ATOMIC_VAR_INIT(0) }

Functions

- static void metal_condition_init (struct metal_condition *cv)
- static int metal_condition_signal (struct metal_condition *cv)
- static int metal_condition_broadcast (struct metal_condition *cv)

8.16.1 Macro Definition Documentation

8.16.1.1 #define METAL_CONDITION_INIT { NULL, ATOMIC_VAR_INIT(0), ATOMIC_VAR_INIT(0) }

Static metal condition variable initialization.

8.16.2 Function Documentation

8.16.2.1 static int metal_condition_broadcast (struct metal_condition * *cv*) [inline], [static]

8.16.2.2 static void metal_condition_init (struct metal_condition * cv) [inline], [static]

8.16.2.3 static int metal_condition_signal (struct metal_condition * cv) [inline], [static]

8.17 lib/config.h File Reference

Macros

- #define METAL_VER_MAJOR @PROJECT_VER_MAJOR@
- #define METAL_VER_MINOR @PROJECT_VER_MINOR@
- #define METAL_VER_PATCH @PROJECT_VER_PATCH@
- #define METAL_VER "@PROJECT_VER@"
- #define METAL_SYSTEM "@PROJECT_SYSTEM@"
- #define METAL_SYSTEM_ @PROJECT_SYSTEM_UPPER@
- #define METAL_PROCESSOR "@PROJECT_PROCESSOR@"
- #define METAL_PROCESSOR_ @PROJECT_PROCESSOR_UPPER@
- #define METAL_MACHINE "@PROJECT_MACHINE@"
- #define METAL_MACHINE_ @PROJECT_MACHINE_UPPER@
- #define HAVE_STDATOMIC_H
- #define HAVE_FUTEX_H

8.17.1 Macro Definition Documentation

8.17.1.1 #define HAVE_FUTEX_H

8.17.1.2 #define HAVE_STDATOMIC_H

8.17.1.3 #define METAL_MACHINE "@PROJECT_MACHINE@"

Machine type (zynq, zynqmp, ...).

8.17.1.4 #define METAL_MACHINE_ @PROJECT_MACHINE_UPPER@

8.17.1.5 #define METAL_PROCESSOR "@PROJECT_PROCESSOR@"

Processor type (arm, x86_64, ...).

8.17.1.6 #define METAL_PROCESSOR_ @PROJECT_PROCESSOR_UPPER@

8.17.1.7 #define METAL_SYSTEM "@PROJECT_SYSTEM@"

System type (linux, generic, ...).

8.17.1.8 #define METAL_SYSTEM_ @PROJECT_SYSTEM_UPPER@

8.17.1.9 #define METAL_VER "@PROJECT_VER@"

Library version string.

8.17.1.10 #define METAL_VER_MAJOR @PROJECT_VER_MAJOR@

Library major version number.

8.17.1.11 #define METAL_VER_MINOR @PROJECT_VER_MINOR@

Library minor version number.

8.17.1.12 #define METAL_VER_PATCH @PROJECT_VER_PATCH@

Library patch level.

8.18 lib/cpu.h File Reference

```
#include <metal/processor/@PROJECT_PROCESSOR@/cpu.h>
```

8.19 lib/processor/aarch64/cpu.h File Reference

Macros

- #define metal_cpu_yield() asm volatile("yield")

8.19.1 Macro Definition Documentation

8.19.1.1 #define metal_cpu_yield( ) asm volatile("yield")

8.20 lib/processor/arm/cpu.h File Reference

Macros

- #define metal_cpu_yield()

8.20.1 Macro Definition Documentation

8.20.1.1 #define metal_cpu_yield( )

8.21 lib/processor/x86_64/cpu.h File Reference

Macros

- #define metal_cpu_yield() asm volatile("rep; nop")

8.21.1 Macro Definition Documentation

8.21.1.1 #define metal_cpu_yield( ) asm volatile("rep; nop")

8.22 lib/device.c File Reference

```
#include <string.h>
#include <errno.h>
#include "metal/device.h"
#include "metal/list.h"
#include "metal/log.h"
#include "metal/sys.h"
#include "metal/utilities.h"
#include "metal/dma.h"
#include "metal/cache.h"
```

Functions

- int metal_bus_register (struct metal_bus *bus)

  Register a libmetal bus.
- int metal_bus_unregister (struct metal_bus *bus)

  Unregister a libmetal bus.
- int metal_bus_find (const char *name, struct metal_bus **result)

  Find a libmetal bus by name.
- int metal_device_open (const char *bus_name, const char *dev_name, struct metal_device **device)

  Open a libmetal device by name.
- void metal_device_close (struct metal_device *device)

  Close a libmetal device.
- int metal_register_generic_device (struct metal_device *device)

  Statically register a generic libmetal device.
- static int metal_generic_dev_open (struct metal_bus *bus, const char *dev_name, struct metal_device **device)
- static int metal_generic_dev_dma_map (struct metal_bus *bus, struct metal_device *device, uint32_t dir, struct metal_sg *sg_in, int nents_in, struct metal_sg *sg_out)
- void metal_generic_dev_dma_unmap (struct metal_bus *bus, struct metal_device *device, uint32_t dir, struct metal_sg *sg, int nents)

Variables

- struct metal_bus metal_generic_bus

8.22.1 Function Documentation

8.22.1.1 static int metal_generic_dev_dma_map (struct metal_bus * *bus*, struct metal_device * *device*, uint32_t *dir*, struct metal_sg * *sg_in*, int *nents_in*, struct metal_sg * *sg_out*) [static]

8.22.1.2 void metal_generic_dev_dma_unmap (struct metal_bus * *bus*, struct metal_device * *device*, uint32_t *dir*, struct metal_sg * *sg*, int *nents*)

8.22.1.3 static int metal_generic_dev_open (struct metal_bus * *bus*, const char * *dev_name*, struct metal_device ** *device*) [static]

8.23 lib/system/linux/device.c File Reference

```
#include "metal/device.h"
#include "metal/sys.h"
#include "metal/utilities.h"
#include "metal/irq.h"
```

Data Structures

- struct linux_driver
- struct linux_bus
- struct linux_device

Macros

- #define MAX_DRIVERS 64
- #define for_each_linux_bus(lbus) for ((lbus) = linux_bus; (lbus)->bus_name; (lbus)++)
- #define for_each_linux_driver(lbus, ldrv) for ((ldrv) = lbus->drivers; (ldrv)->drv_name; (ldrv)++)

Functions

- static struct linux_bus * to_linux_bus (struct metal_bus *bus)
- static struct linux_device * to_linux_device (struct metal_device *device)
- static int metal_uio_read_map_attr (struct linux_device *ldev, unsigned index, const char *name, unsigned long *value)
- static int metal_uio_dev_bind (struct linux_device *ldev, struct linux_driver *ldrv)
- static int metal_uio_dev_open (struct linux_bus *lbus, struct linux_device *ldev)
- static void metal_uio_dev_close (struct linux_bus *lbus, struct linux_device *ldev)
- static void metal_uio_dev_irq_ack (struct linux_bus *lbus, struct linux_device *ldev, int irq)
- static int metal_uio_dev_dma_map (struct linux_bus *lbus, struct linux_device *ldev, uint32_t dir, struct metal_sg *sg_in, int nents_in, struct metal_sg *sg_out)
- static void metal_uio_dev_dma_unmap (struct linux_bus *lbus, struct linux_device *ldev, uint32_t dir, struct metal_sg *sg, int nents)
- static int metal_linux_dev_open (struct metal_bus *bus, const char *dev_name, struct metal_device **device)
- static void metal_linux_dev_close (struct metal_bus *bus, struct metal_device *device)
- static void metal_linux_bus_close (struct metal_bus *bus)
- static void metal_linux_dev_irq_ack (struct metal_bus *bus, struct metal_device *device, int irq)
- static int metal_linux_dev_dma_map (struct metal_bus *bus, struct metal_device *device, uint32_t dir, struct metal_sg *sg_in, int nents_in, struct metal_sg *sg_out)
- static void metal_linux_dev_dma_unmap (struct metal_bus *bus, struct metal_device *device, uint32_t dir, struct metal_sg *sg, int nents)
- static int metal_linux_register_bus (struct linux_bus *lbus)
- static int metal_linux_probe_driver (struct linux_bus *lbus, struct linux_driver *ldrv)
- static int metal_linux_probe_bus (struct linux_bus *lbus)
- int metal_linux_bus_init (void)
- void metal_linux_bus_finish (void)

Variables

- static struct linux_bus linux_bus []
- static const struct metal_bus_ops metal_linux_bus_ops

8.23.1 Macro Definition Documentation

8.23.1.1 #define for_each_linux_bus(*lbus*) for ((lbus) = linux_bus; (lbus)->bus_name; (lbus)++)

8.23.1.2 #define for_each_linux_driver(*lbus, ldrv*) for ((ldrv) = lbus->drivers; (ldrv)->drv_name; (ldrv)++)

8.23.1.3 #define MAX_DRIVERS 64

8.23.2 Function Documentation

8.23.2.1 static void metal_linux_bus_close (struct metal_bus * *bus*) [static]

8.23.2.2 void metal_linux_bus_finish (void)

8.23.2.3 int metal_linux_bus_init (void)

8.23.2.4 static void metal_linux_dev_close (struct metal_bus * *bus,* struct metal_device * *device*) [static]

8.23.2.5 static int metal_linux_dev_dma_map (struct metal_bus * *bus,* struct metal_device * *device,* uint32_t *dir,* struct metal_sg * *sg_in,* int *nents_in,* struct metal_sg * *sg_out*) [static]

8.23.2.6 static void metal_linux_dev_dma_unmap (struct metal_bus * *bus,* struct metal_device * *device,* uint32_t *dir,* struct metal_sg * *sg,* int *nents*) [static]

8.23.2.7 static void metal_linux_dev_irq_ack (struct metal_bus * *bus,* struct metal_device * *device,* int *irq*) [static]

8.23.2.8 static int metal_linux_dev_open (struct metal_bus * *bus,* const char * *dev_name,* struct metal_device ** *device*) [static]

8.23.2.9 static int metal_linux_probe_bus (struct linux_bus * *lbus*) [static]

8.23.2.10 static int metal_linux_probe_driver (struct linux_bus * *lbus,* struct linux_driver * *ldrv*) [static]

8.23.2.11 static int metal_linux_register_bus (struct linux_bus * *lbus*) [static]

8.23.2.12 static int metal_uio_dev_bind (struct linux_device * *ldev,* struct linux_driver * *ldrv*) [static]

8.23.2.13 static void metal_uio_dev_close (struct linux_bus * *lbus,* struct linux_device * *ldev*) [static]

8.23.2.14 static int metal_uio_dev_dma_map (struct linux_bus * *lbus,* struct linux_device * *ldev,* uint32_t *dir,* struct metal_sg * *sg_in,* int *nents_in,* struct metal_sg * *sg_out*) [static]

8.23.2.15 static void metal_uio_dev_dma_unmap (struct linux_bus * *lbus,* struct linux_device * *ldev,* uint32_t *dir,* struct metal_sg * *sg,* int *nents*) [static]

8.23.2.16 static void metal_uio_dev_irq_ack (struct linux_bus * *lbus,* struct linux_device * *ldev,* int *irq*) [static]

8.23.2.17 static int metal_uio_dev_open (struct linux_bus * *lbus,* struct linux_device * *ldev*) [static]

8.23.2.18 static int metal_uio_read_map_attr (struct linux_device * *ldev,* unsigned *index,* const char * *name,* unsigned long * *value*) [static]

8.23.2.19 static struct linux_bus* to_linux_bus (struct metal_bus * *bus*) [static]

8.23.2.20 static struct linux_device* to_linux_device (struct metal_device * *device*) [static]

8.23.3 Variable Documentation

8.23.3.1 struct linux_bus linux_bus[] [static]

8.23.3.2 const struct metal_bus_ops metal_linux_bus_ops [static]

Initial value:

```
= {
        .bus_close      = metal_linux_bus_close,
        .dev_open       = metal_linux_dev_open,
        .dev_close      = metal_linux_dev_close,
        .dev_irq_ack    = metal_linux_dev_irq_ack,
        .dev_dma_map    = metal_linux_dev_dma_map,
        .dev_dma_unmap  = metal_linux_dev_dma_unmap,
}
```

8.24 lib/device.h File Reference

```
#include <stdint.h>
#include "metal/io.h"
#include "metal/list.h"
#include "metal/dma.h"
#include "metal/sys.h"
```

Data Structures

- struct metal_bus_ops
- struct metal_bus
- struct metal_device

Macros

- #define METAL_MAX_DEVICE_REGIONS 32

Functions

- int metal_bus_register (struct metal_bus *bus)

  Register a libmetal bus.
- int metal_bus_unregister (struct metal_bus *bus)

  Unregister a libmetal bus.
- int metal_bus_find (const char *name, struct metal_bus **bus)

  Find a libmetal bus by name.
- int metal_register_generic_device (struct metal_device *device)

  Statically register a generic libmetal device.
- int metal_device_open (const char *bus_name, const char *dev_name, struct metal_device **device)

  Open a libmetal device by name.
- void metal_device_close (struct metal_device *device)

  Close a libmetal device.
- static struct metal_io_region * metal_device_io_region (struct metal_device *device, unsigned index)

  Get an I/O region accessor for a device region.

Variables

- struct metal_bus metal_generic_bus

8.25 lib/dma.c File Reference

```
#include <errno.h>
#include <string.h>
#include "metal/device.h"
#include "metal/log.h"
#include "metal/dma.h"
#include "metal/atomic.h"
```

Functions

- int metal_dma_map (struct metal_device *dev, uint32_t dir, struct metal_sg *sg_in, int nents_in, struct metal_sg *sg_out)

  Map memory for DMA transaction. After the memory is DMA mapped, the memory should be accessed by the DMA device but not the CPU.
- void metal_dma_unmap (struct metal_device *dev, uint32_t dir, struct metal_sg *sg, int nents)

  Unmap DMA memory After the memory is DMA unmapped, the memory should be accessed by the CPU but not the DMA device.

8.26 lib/dma.h File Reference

```
#include <stdint.h>
#include "metal/sys.h"
```

Data Structures

- struct metal_sg

  scatter/gather list element structure

Macros

- #define METAL_DMA_DEV_R 1
- #define METAL_DMA_DEV_W 2
- #define METAL_DMA_DEV_WR 3

Functions

- int metal_dma_map (struct metal_device *dev, uint32_t dir, struct metal_sg *sg_in, int nents_in, struct metal_sg *sg_out)

  Map memory for DMA transaction. After the memory is DMA mapped, the memory should be accessed by the DMA device but not the CPU.
- void metal_dma_unmap (struct metal_device *dev, uint32_t dir, struct metal_sg *sg, int nents)

  Unmap DMA memory After the memory is DMA unmapped, the memory should be accessed by the CPU but not the DMA device.

8.27 lib/init.c File Reference

```
#include <string.h>
#include "metal/sys.h"
```

Functions

- int metal_init (const struct metal_init_params *params)

  Initialize libmetal.
- void metal_finish (void)

  Shutdown libmetal.

8.28 lib/system/freertos/init.c File Reference

```
#include "metal/sys.h"
#include "metal/utilities.h"
```

Functions

- int metal_irq_init (void)
- void metal_irq_deinit (void)
- int metal_sys_init (const struct metal_init_params *params)
- void metal_sys_finish (void)

Variables

- struct metal_state _metal

8.28.1 Function Documentation

8.28.1.1 void metal_irq_deinit (void)

8.28.1.2 int metal_irq_init (void)

8.28.1.3 void metal_sys_finish (void)

8.28.1.4 int metal_sys_init (const struct metal_init_params * *params*)

8.29 lib/system/generic/init.c File Reference

```
#include "metal/sys.h"
#include "metal/utilities.h"
```

Functions

- int metal_irq_init (void)
- void metal_irq_deinit (void)

- int metal_sys_init (const struct metal_init_params *params)
- void metal_sys_finish (void)

Variables

- struct metal_state _metal

8.29.1 Function Documentation

8.29.1.1 void metal_irq_deinit (void)

8.29.1.2 int metal_irq_init (void)

8.29.1.3 void metal_sys_finish (void)

8.29.1.4 int metal_sys_init (const struct metal_init_params * *params*)

8.30 lib/system/linux/init.c File Reference

```
#include <sys/types.h>
#include "metal/sys.h"
#include "metal/utilities.h"
```

Functions

- int metal_linux_irq_init ()

  irq handling initialization
- void metal_linux_irq_shutdown ()

  irq handling shutdown
- static int metal_pagesize_compare (const void *_a, const void *_b)
- static int metal_add_page_size (const char *path, int shift, int mmap_flags)
- static int metal_init_page_sizes (void)
- int metal_sys_init (const struct metal_init_params *params)
- void metal_sys_finish (void)

Variables

- struct metal_state _metal

8.30.1 Function Documentation

8.30.1.1 static int metal_add_page_size (const char * *path*, int *shift*, int *mmap_flags*) [static]

8.30.1.2 static int metal_init_page_sizes (void) [static]

8.30.1.3 int metal_linux_irq_init ( )

irq handling initialization

Returns

0 on sucess, non-zero on failure

8.30.1.4 void metal_linux_irq_shutdown ( )

irq handling shutdown

8.30.1.5 static int metal_pagesize_compare (const void * *_a*, const void * *_b*) [static]

Sort function for page size array.

8.30.1.6 void metal_sys_finish (void)

8.30.1.7 int metal_sys_init (const struct **metal_init_params** * *params*)

8.31 lib/io.c File Reference

```
#include "metal/io.h"
```

Functions

- void * metal_generic_memset_io (void *dst, int c, size_t size)
- void * metal_generic_memcpy_io (void *dst, const void *src, size_t size)
- void * metal_memset_io (void *dst, int c, size_t size)

  libmetal set device memory
- void * metal_memcpy_io (void *dst, const void *src, size_t size)

  libmetal copy to target memory

8.31.1 Function Documentation

8.31.1.1 void* metal_generic_memcpy_io (void * *dst*, const void * *src*, size_t *size*)

8.31.1.2 void* metal_generic_memset_io (void * *dst*, int *c*, size_t *size*)

8.32 lib/system/freertos/io.c File Reference

```
#include "metal/io.h"
```

Functions

- void metal_machine_io_mem_map (metal_phys_addr_t pa, size_t size, unsigned int flags)
- void * metal_io_mem_map (metal_phys_addr_t pa, struct metal_io_region *io, size_t size)

  libmetal memory map

8.32.1 Function Documentation

8.32.1.1 void metal_machine_io_mem_map (**metal_phys_addr_t** *pa*, size_t *size*, unsigned int *flags*)

8.33 lib/system/generic/io.c File Reference

```
#include "metal/io.h"
```

Functions

- void metal_machine_io_mem_map (metal_phys_addr_t pa, size_t size, unsigned int flags)
- void * metal_io_mem_map (metal_phys_addr_t pa, struct metal_io_region *io, size_t size)

  libmetal memory map

8.33.1 Function Documentation

8.33.1.1 void metal_machine_io_mem_map (metal_phys_addr_t *pa,* size_t *size,* unsigned int *flags*)

8.34 lib/system/linux/io.c File Reference

```
#include "metal/io.h"
```

Functions

- void * metal_io_mem_map (metal_phys_addr_t pa, struct metal_io_region *io, size_t size)

  libmetal memory map

8.35 lib/io.h File Reference

```
#include <assert.h>
#include <stdint.h>
#include <string.h>
#include <stdlib.h>
#include "metal/atomic.h"
#include "metal/sys.h"
```

Data Structures

- struct metal_io_ops
- struct metal_io_region

Macros

- #define METAL_CACHE_UNKNOWN 0x0 /** Use cache, unknown cache type */
- #define METAL_UNCACHED 0x1 /** No cache */
- #define METAL_CACHE_WB 0x2 /** Write back */
- #define METAL_CACHE_WT 0x4 /** Write through */
- #define METAL_MEM_MAPPED 0x10 /** Memory mapped */
- #define METAL_IO_MAPPED 0x20 /** I/O mapped */
- #define METAL_SHARED_MEM 0x40 /** Shared memory */
- #define METAL_DMA_NO_CACHE_OPS 0x0 /** No cache ops in DMA transaction */

- #define METAL_DMA_CACHE_OPS 0x1 /** Require cache ops in DMA transaction */
- #define metal_io_read8_explicit(_io, _ofs, _order) metal_io_read((_io), (_ofs), (_order), 1)
- #define metal_io_read8(_io, _ofs) metal_io_read((_io), (_ofs), memory_order_seq_cst, 1)
- #define metal_io_write8_explicit(_io, _ofs, _val, _order) metal_io_write((_io), (_ofs), (_val), (_order), 1)
- #define metal_io_write8(_io, _ofs, _val) metal_io_write((_io), (_ofs), (_val), memory_order_seq_cst, 1)
- #define metal_io_read16_explicit(_io, _ofs, _order) metal_io_read((_io), (_ofs), (_order), 2)
- #define metal_io_read16(_io, _ofs) metal_io_read((_io), (_ofs), memory_order_seq_cst, 2)
- #define metal_io_write16_explicit(_io, _ofs, _val, _order) metal_io_write((_io), (_ofs), (_val), (_order), 2)
- #define metal_io_write16(_io, _ofs, _val) metal_io_write((_io), (_ofs), (_val), memory_order_seq_cst, 2)
- #define metal_io_read32_explicit(_io, _ofs, _order) metal_io_read((_io), (_ofs), (_order), 4)
- #define metal_io_read32(_io, _ofs) metal_io_read((_io), (_ofs), memory_order_seq_cst, 4)
- #define metal_io_write32_explicit(_io, _ofs, _val, _order) metal_io_write((_io), (_ofs), (_val), (_order), 4)
- #define metal_io_write32(_io, _ofs, _val) metal_io_write((_io), (_ofs), (_val), memory_order_seq_cst, 4)
- #define metal_io_read64_explicit(_io, _ofs, _order) metal_io_read((_io), (_ofs), (_order), 8)
- #define metal_io_read64(_io, _ofs) metal_io_read((_io), (_ofs), memory_order_seq_cst, 8)
- #define metal_io_write64_explicit(_io, _ofs, _val, _order) metal_io_write((_io), (_ofs), (_val), (_order), 8)
- #define metal_io_write64(_io, _ofs, _val) metal_io_write((_io), (_ofs), (_val), memory_order_seq_cst, 8)

Functions

- static void metal_io_init (struct metal_io_region *io, void *virt, const metal_phys_addr_t *physmap, size_t size, unsigned page_shift, unsigned int mem_flags, const struct metal_io_ops *ops)

  Open a libmetal I/O region.
- static void metal_io_finish (struct metal_io_region *io)

  Close a libmetal shared memory segment.
- static size_t metal_io_region_size (struct metal_io_region *io)

  Get size of I/O region.
- static void * metal_io_virt (struct metal_io_region *io, unsigned long offset)

  Get virtual address for a given offset into the I/O region.
- static unsigned long metal_io_virt_to_offset (struct metal_io_region *io, void *virt)

  Convert a virtual address to offset within I/O region.
- static metal_phys_addr_t metal_io_phys (struct metal_io_region *io, unsigned long offset)

  Get physical address for a given offset into the I/O region.
- static unsigned long metal_io_phys_to_offset (struct metal_io_region *io, metal_phys_addr_t phys)

  Convert a physical address to offset within I/O region.
- static void * metal_io_phys_to_virt (struct metal_io_region *io, metal_phys_addr_t phys)

  Convert a physical address to virtual address.
- static metal_phys_addr_t metal_io_virt_to_phys (struct metal_io_region *io, void *virt)

  Convert a virtual address to physical address.
- static uint64_t metal_io_read (struct metal_io_region *io, unsigned long offset, memory_order order, int width)

  Read a value from an I/O region.
- static void metal_io_write (struct metal_io_region *io, unsigned long offset, uint64_t value, memory_order order, int width)

  Write a value into an I/O region.
- void * metal_io_mem_map (metal_phys_addr_t pa, struct metal_io_region *io, size_t size)

  libmetal memory map
- void * metal_memset_io (void *dst, int c, size_t size)

  libmetal set device memory
- void * metal_memcpy_io (void *dst, const void *src, size_t size)

  libmetal copy to target memory

8.36 lib/irq.h File Reference

```
#include <stdlib.h>
#include "metal/system/@PROJECT_SYSTEM@/irq.h"
```

Macros

- #define METAL_IRQ_NOT_HANDLED 0
- #define METAL_IRQ_HANDLED 1

Typedefs

- typedef int(∗ metal_irq_handler)(int irq, void ∗priv)

  type of interrupt handler

Functions

- int metal_irq_register (int irq, metal_irq_handler irq_handler, struct metal_device ∗dev, void ∗drv_id)

  register interrupt register interrupt handling of a specific interrupt.
- unsigned int metal_irq_save_disable (void)

  disable interrupts
- void metal_irq_restore_enable (unsigned int flags)

  restore interrupts to their previous state
- void metal_irq_enable (unsigned int vector)

  metal_irq_enable
- void metal_irq_disable (unsigned int vector)

  metal_irq_disable

8.37 lib/system/freertos/irq.h File Reference

Functions

- void metal_irq_isr (unsigned int vector)

  default interrupt handler

8.37.1 Function Documentation

8.37.1.1 void metal_irq_isr (unsigned int *vector*)

default interrupt handler

Parameters

in	*interrupt*	vector

default interrupt handler < irq handler

< irq handler

8.38 lib/system/generic/irq.h File Reference

Functions

- void metal_irq_isr (unsigned int vector)

  default interrupt handler

8.38.1 Function Documentation

8.38.1.1 void metal_irq_isr (unsigned int *vector*)

default interrupt handler

Parameters

in	*interrupt*	vector

default interrupt handler < irq handler

< irq handler

8.39 lib/system/linux/irq.h File Reference

8.40 lib/list.h File Reference

```
#include <stdlib.h>
```

Data Structures

- struct metal_list

Macros

- #define METAL_DECLARE_LIST(name) struct metal_list name = { .next = &name, .prev = &name }
- #define metal_list_for_each(list, node)

Functions

- static void metal_list_init (struct metal_list *list)
- static void metal_list_add_before (struct metal_list *node, struct metal_list *new_node)
- static void metal_list_add_after (struct metal_list *node, struct metal_list *new_node)
- static void metal_list_add_head (struct metal_list *list, struct metal_list *node)
- static void metal_list_add_tail (struct metal_list *list, struct metal_list *node)
- static int metal_list_is_empty (struct metal_list *list)
- static void metal_list_del (struct metal_list *node)
- static struct metal_list * metal_list_first (struct metal_list *list)

8.41 lib/log.c File Reference

```
#include <stdarg.h>
#include <stdio.h>
#include "metal/log.h"
#include "metal/sys.h"
```

Functions

- void metal_default_log_handler (enum metal_log_level level, const char *format,...)

  Default libmetal log handler. This handler prints libmetal log mesages to stderr.
- void metal_set_log_handler (metal_log_handler handler)

  Set libmetal log handler.
- metal_log_handler metal_get_log_handler (void)

  Get the current libmetal log handler.
- void metal_set_log_level (enum metal_log_level level)

  Set the level for libmetal logging.
- enum metal_log_level metal_get_log_level (void)

  Get the current level for libmetal logging.

8.42 lib/log.h File Reference

Macros

- #define metal_log(level,...)

Typedefs

- typedef void(* metal_log_handler)(enum metal_log_level level, const char *format,...)

Enumerations

- enum metal_log_level {
  LOG_EMERGENCY, LOG_ALERT, LOG_CRITICAL, LOG_ERROR,
  LOG_WARNING, LOG_NOTICE, LOG_INFO, LOG_DEBUG }

Functions

- void metal_set_log_handler (metal_log_handler handler)

  Set libmetal log handler.
- metal_log_handler metal_get_log_handler (void)

  Get the current libmetal log handler.
- void metal_set_log_level (enum metal_log_level level)

  Set the level for libmetal logging.
- enum metal_log_level metal_get_log_level (void)

  Get the current level for libmetal logging.
- void metal_default_log_handler (enum metal_log_level level, const char *format,...)

  Default libmetal log handler. This handler prints libmetal log mesages to stderr.

8.43 lib/mutex.h File Reference

```
#include "metal/system/@PROJECT_SYSTEM@/mutex.h"
```

Functions

- static void metal_mutex_init (metal_mutex_t *mutex)

  Initialize a libmetal mutex.
- static void metal_mutex_deinit (metal_mutex_t *mutex)

  Deinitialize a libmetal mutex.
- static int metal_mutex_try_acquire (metal_mutex_t *mutex)

  Try to acquire a mutex.
- static void metal_mutex_acquire (metal_mutex_t *mutex)

  Acquire a mutex.
- static void metal_mutex_release (metal_mutex_t *mutex)

  Release a previously acquired mutex.
- static int metal_mutex_is_acquired (metal_mutex_t *mutex)

  Checked if a mutex has been acquired.

8.44 lib/system/freertos/mutex.h File Reference

```
#include <assert.h>
#include "FreeRTOS.h"
#include "semphr.h"
```

Data Structures

- struct metal_mutex_t

Macros

- #define METAL_MUTEX_INIT { NULL }

Functions

- static void metal_mutex_init (metal_mutex_t *mutex)
- static void metal_mutex_deinit (metal_mutex_t *mutex)
- static int metal_mutex_try_acquire (metal_mutex_t *mutex)
- static void metal_mutex_acquire (metal_mutex_t *mutex)
- static void metal_mutex_release (metal_mutex_t *mutex)
- static int metal_mutex_is_acquired (metal_mutex_t *mutex)

8.44.1 Macro Definition Documentation

8.44.1.1 #define METAL_MUTEX_INIT { NULL }

8.44.2 Function Documentation

8.44.2.1 static void metal_mutex_acquire (metal_mutex_t * *mutex*) [inline], [static]

8.44.2.2 static void metal_mutex_deinit (metal_mutex_t * *mutex*) [inline], [static]

8.44.2.3 static void metal_mutex_init (metal_mutex_t * *mutex*) [inline], [static]

8.44.2.4 static int metal_mutex_is_acquired (metal_mutex_t * *mutex*) [inline], [static]

8.44.2.5 static void metal_mutex_release (metal_mutex_t * *mutex*) [inline], [static]

8.44.2.6 static int metal_mutex_try_acquire (metal_mutex_t * *mutex*) [inline], [static]

8.45 lib/system/generic/mutex.h File Reference

```
#include "metal/atomic.h"
```

Data Structures

- struct metal_mutex_t

Macros

- #define METAL_MUTEX_INIT { ATOMIC_VAR_INIT(0) }

Functions

- static void metal_mutex_init (metal_mutex_t *mutex)
- static void metal_mutex_deinit (metal_mutex_t *mutex)
- static int metal_mutex_try_acquire (metal_mutex_t *mutex)
- static void metal_mutex_acquire (metal_mutex_t *mutex)
- static void metal_mutex_release (metal_mutex_t *mutex)
- static int metal_mutex_is_acquired (metal_mutex_t *mutex)

8.45.1 Macro Definition Documentation

8.45.1.1 #define METAL_MUTEX_INIT { ATOMIC_VAR_INIT(0) }

8.45.2 Function Documentation

8.45.2.1 static void metal_mutex_acquire (metal_mutex_t * *mutex*) [inline], [static]

8.45.2.2 static void metal_mutex_deinit (metal_mutex_t * *mutex*) [inline], [static]

8.45.2.3 static void metal_mutex_init (metal_mutex_t * *mutex*) [inline], [static]

8.45.2.4 static int metal_mutex_is_acquired (metal_mutex_t * *mutex*) [inline], [static]

8.45.2.5 static void metal_mutex_release (metal_mutex_t * *mutex*) [inline], [static]

8.45.2.6 static int metal_mutex_try_acquire (metal_mutex_t * *mutex*) [inline], [static]

8.46 lib/system/linux/mutex.h File Reference

```
#include <unistd.h>
#include <sys/syscall.h>
#include <linux/futex.h>
#include "metal/atomic.h"
```

Data Structures

- struct metal_mutex_t

Macros

- #define METAL_MUTEX_INIT { ATOMIC_VAR_INIT(0) }

Functions

- static int __metal_mutex_cmpxchg (metal_mutex_t *mutex, int exp, int val)
- static void metal_mutex_init (metal_mutex_t *mutex)
- static void metal_mutex_deinit (metal_mutex_t *mutex)
- static int metal_mutex_try_acquire (metal_mutex_t *mutex)
- static void metal_mutex_acquire (metal_mutex_t *mutex)
- static void metal_mutex_release (metal_mutex_t *mutex)
- static int metal_mutex_is_acquired (metal_mutex_t *mutex)

8.46.1 Macro Definition Documentation

8.46.1.1 #define METAL_MUTEX_INIT { ATOMIC_VAR_INIT(0) }

8.46.2 Function Documentation

8.46.2.1 static int __metal_mutex_cmpxchg (metal_mutex_t * *mutex,* int *exp,* int *val*) [inline], [static]

8.46.2.2 static void metal_mutex_acquire (metal_mutex_t * *mutex*) [inline], [static]

8.46.2.3 static void metal_mutex_deinit (metal_mutex_t * *mutex*) [inline], [static]

8.46.2.4 static void metal_mutex_init (metal_mutex_t * *mutex*) [inline], [static]

8.46.2.5 static int metal_mutex_is_acquired (metal_mutex_t * *mutex*) [inline], [static]

8.46.2.6 static void metal_mutex_release (metal_mutex_t * *mutex*) [inline], [static]

8.46.2.7 static int metal_mutex_try_acquire (metal_mutex_t * *mutex*) [inline], [static]

8.47 lib/shmem.c File Reference

```
#include <errno.h>
#include "metal/shmem.h"
#include "metal/sys.h"
#include "metal/utilities.h"
```

Functions

- int metal_shmem_register_generic (struct metal_generic_shmem *shmem)

  Statically register a generic shared memory region.
- int metal_shmem_open_generic (const char *name, size_t size, struct metal_io_region **result)

8.47.1 Function Documentation

8.47.1.1 int metal_shmem_open_generic (const char * *name*, size_t *size*, struct metal_io_region ** *result*)

8.48 lib/system/freertos/shmem.c File Reference

```
#include "metal/shmem.h"
```

Functions

- int metal_shmem_open (const char *name, size_t size, struct metal_io_region **io)

  Open a libmetal shared memory segment.

8.49 lib/system/generic/shmem.c File Reference

```
#include "metal/shmem.h"
```

Functions

- int metal_shmem_open (const char *name, size_t size, struct metal_io_region **io)

  Open a libmetal shared memory segment.

8.50 lib/system/linux/shmem.c File Reference

```
#include "metal/shmem.h"
#include "metal/sys.h"
#include "metal/utilities.h"
```

Data Structures

- struct metal_shmem

Functions

- static void metal_shmem_io_close (struct metal_io_region *io)
- static int metal_shmem_try_map (struct metal_page_size *ps, int fd, size_t size, struct metal_io_region **result)
- int metal_shmem_open (const char *name, size_t size, struct metal_io_region **result)

  Open a libmetal shared memory segment.

Variables

- static const struct metal_io_ops metal_shmem_io_ops

8.50.1 Function Documentation

8.50.1.1 static void metal_shmem_io_close (struct metal_io_region * *io*) [static]

8.50.1.2 static int metal_shmem_try_map (struct metal_page_size * *ps*, int *fd*, size_t *size*, struct metal_io_region ** *result*) [static]

8.50.2 Variable Documentation

8.50.2.1 const struct metal_io_ops metal_shmem_io_ops [static]

Initial value:

```
= {
        NULL, NULL, metal_shmem_io_close
}
```

8.51 lib/shmem.h File Reference

```
#include "metal/io.h"
```

Data Structures

- struct metal_generic_shmem

Functions

- int metal_shmem_open (const char *name, size_t size, struct metal_io_region **io)

  Open a libmetal shared memory segment.
- int metal_shmem_register_generic (struct metal_generic_shmem *shmem)

  Statically register a generic shared memory region.

8.52 lib/sleep.h File Reference

Functions

- int metal_sleep_usec (unsigned int usec)

  delay in microseconds delay the next execution in the calling thread fo usec microseconds.

8.53 lib/spinlock.h File Reference

```
#include "metal/atomic.h"
#include "metal/cpu.h"
```

Data Structures

- struct metal_spinlock

Macros

- #define METAL_SPINLOCK_INIT {ATOMIC_VAR_INIT(0)}

Functions

- static void metal_spinlock_init (struct metal_spinlock *slock)

  Initialize a libmetal spinlock.
- static void metal_spinlock_acquire (struct metal_spinlock *slock)

  Acquire a spinlock.
- static void metal_spinlock_release (struct metal_spinlock *slock)

  Release a previously acquired spinlock.

8.54 lib/sys.h File Reference

```
#include <stdlib.h>
#include "metal/log.h"
#include "metal/list.h"
#include "metal/system/@PROJECT_SYSTEM@/sys.h"
```

Data Structures

- struct metal_init_params
- struct metal_common_state

Macros

- #define METAL_BAD_OFFSET ((unsigned long)-1)
- #define METAL_BAD_PHYS ((metal_phys_addr_t)-1)
- #define METAL_BAD_IRQ ((metal_irq_t)-1)
- #define METAL_INIT_DEFAULTS

Typedefs

- typedef unsigned long metal_phys_addr_t
- typedef int metal_irq_t

Functions

- int metal_init (const struct metal_init_params *params)

  Initialize libmetal.
- void metal_finish (void)

  Shutdown libmetal.

Variables

- struct metal_state _metal

8.55 lib/system/freertos/sys.h File Reference

```
#include "./@PROJECT_MACHINE@/sys.h"
```

Data Structures

- struct metal_state

Macros

- #define METAL_MAX_DEVICE_REGIONS 1

8.55.1 Macro Definition Documentation

8.55.1.1 #define METAL_MAX_DEVICE_REGIONS 1

8.56 lib/system/freertos/zynq7/sys.h File Reference

```
#include "xscugic.h"
```

Macros

- #define MAX_IRQS ((int)XSCUGIC_MAX_NUM_INTR_INPUTS)

Functions

- static void sys_irq_enable (unsigned int vector)
- static void sys_irq_disable (unsigned int vector)

8.56.1 Macro Definition Documentation

8.56.1.1 #define MAX_IRQS ((int)XSCUGIC_MAX_NUM_INTR_INPUTS)

maximum number of irqs

8.56.2 Function Documentation

8.56.2.1 static void sys_irq_disable (unsigned int *vector*) [inline], [static]

8.56.2.2 static void sys_irq_enable (unsigned int *vector*) [inline], [static]

8.57 lib/system/freertos/zynqmp_r5/sys.h File Reference

```
#include "xscugic.h"
```

Macros

- #define MAX_IRQS ((int)XSCUGIC_MAX_NUM_INTR_INPUTS)

Functions

- static void sys_irq_enable (unsigned int vector)
- static void sys_irq_disable (unsigned int vector)

8.57.1 Macro Definition Documentation

8.57.1.1 #define MAX_IRQS ((int)XSCUGIC_MAX_NUM_INTR_INPUTS)

maximum number of irqs

8.57.2 Function Documentation

8.57.2.1 static void sys_irq_disable (unsigned int *vector*) [inline], [static]

8.57.2.2 static void sys_irq_enable (unsigned int *vector*) [inline], [static]

8.58 lib/system/generic/sys.h File Reference

```
#include <errno.h>
#include <fcntl.h>
#include <libgen.h>
#include <limits.h>
#include <stdio.h>
#include <stdlib.h>
#include <stdarg.h>
#include <string.h>
#include "./@PROJECT_MACHINE@/sys.h"
```

Data Structures

- struct metal_state

Macros

- #define METAL_MAX_DEVICE_REGIONS 1

8.58.1 Macro Definition Documentation

8.58.1.1 #define METAL_MAX_DEVICE_REGIONS 1

8.59 lib/system/generic/zynq7/sys.h File Reference

```
#include "xscugic.h"
```

Macros

- #define MAX_IRQS ((int)XSCUGIC_MAX_NUM_INTR_INPUTS)

Functions

- static void sys_irq_enable (unsigned int vector)
- static void sys_irq_disable (unsigned int vector)

8.59.1 Macro Definition Documentation

8.59.1.1 #define MAX_IRQS ((int)XSCUGIC_MAX_NUM_INTR_INPUTS)

maximum number of irqs

8.59.2 Function Documentation

8.59.2.1 static void sys_irq_disable (unsigned int *vector*) [inline], [static]

8.59.2.2 static void sys_irq_enable (unsigned int *vector*) [inline], [static]

8.60 lib/system/generic/zynqmp_r5/sys.h File Reference

```
#include "xscugic.h"
```

Macros

- #define MAX_IRQS ((int)XSCUGIC_MAX_NUM_INTR_INPUTS)

Functions

- static void sys_irq_enable (unsigned int vector)
- static void sys_irq_disable (unsigned int vector)

8.60.1 Macro Definition Documentation

8.60.1.1 #define MAX_IRQS ((int)XSCUGIC_MAX_NUM_INTR_INPUTS)

maximum number of irqs

8.60.2 Function Documentation

8.60.2.1 static void sys_irq_disable (unsigned int *vector*) [inline], [static]

8.60.2.2 static void sys_irq_enable (unsigned int *vector*) [inline], [static]

8.61 lib/system/linux/sys.h File Reference

```
#include <errno.h>
#include <fcntl.h>
#include <libgen.h>
#include <limits.h>
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#include <stdarg.h>
#include <syslog.h>
#include <sys/file.h>
#include <sys/mman.h>
#include <sys/stat.h>
#include <sys/syscall.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <linux/futex.h>
#include <sysfs/libsysfs.h>
#include <hugetlbfs.h>
```

Data Structures

- struct metal_page_size
- struct metal_state

Macros

- #define METAL_INVALID_VADDR NULL
- #define MAX_PAGE_SIZES 32

8.61.1 Macro Definition Documentation

8.61.1.1 #define MAX_PAGE_SIZES 32

8.61.1.2 #define METAL_INVALID_VADDR NULL

8.62 lib/system/freertos/cache.c File Reference

```
#include "metal/cache.h"
```

Functions

- void metal_machine_cache_flush (void *addr, unsigned int len)

- void metal_machine_cache_invalidate (void *addr, unsigned int len)
- void metal_cache_flush (void *addr, unsigned int len)

  flush specified data cache
- void metal_cache_invalidate (void *addr, unsigned int len)

  invalidate specified data cache

8.62.1 Function Documentation

8.62.1.1 void metal_machine_cache_flush (void * *addr,* unsigned int *len*)

8.62.1.2 void metal_machine_cache_invalidate (void * *addr,* unsigned int *len*)

8.63 lib/system/generic/cache.c File Reference

```
#include "metal/cache.h"
```

Functions

- void metal_machine_cache_flush (void *addr, unsigned int len)
- void metal_machine_cache_invalidate (void *addr, unsigned int len)
- void metal_cache_flush (void *addr, unsigned int len)

  flush specified data cache
- void metal_cache_invalidate (void *addr, unsigned int len)

  invalidate specified data cache

8.63.1 Function Documentation

8.63.1.1 void metal_machine_cache_flush (void * *addr,* unsigned int *len*)

8.63.1.2 void metal_machine_cache_invalidate (void * *addr,* unsigned int *len*)

8.64 lib/system/linux/cache.c File Reference

```
#include "metal/cache.h"
```

Functions

- void metal_cache_flush (void *addr, unsigned int len)

  flush specified data cache
- void metal_cache_invalidate (void *addr, unsigned int len)

  invalidate specified data cache

8.65 lib/system/freertos/condition.c File Reference

```
#include "metal/condition.h"
```

Functions

- int metal_condition_wait (struct metal_condition *cv, metal_mutex_t *m)

  Block until the condition variable is notified. Before calling this function, the caller should have acquired the mutex.

8.66 lib/system/generic/condition.c File Reference

```
#include "metal/condition.h"
#include "metal/irq.h"
```

Functions

- void metal_generic_default_poll (void)
- int metal_condition_wait (struct metal_condition *cv, metal_mutex_t *m)

  Block until the condition variable is notified. Before calling this function, the caller should have acquired the mutex.

8.66.1 Function Documentation

8.66.1.1 void metal_generic_default_poll (void)

8.67 lib/system/linux/condition.c File Reference

```
#include "metal/condition.h"
```

Functions

- int metal_condition_wait (struct metal_condition *cv, metal_mutex_t *m)

  Block until the condition variable is notified. Before calling this function, the caller should have acquired the mutex.

8.68 lib/system/freertos/irq.c File Reference

```
#include <errno.h>
#include "metal/irq.h"
#include "metal/sys.h"
#include "metal/log.h"
#include "metal/mutex.h"
```

Data Structures

- struct metal_irq_hddesc
- struct metal_irqs_state

Macros

- #define MAX_HDS 10

Functions

- int metal_irq_register (int irq, metal_irq_handler hd, struct metal_device *dev, void *drv_id)
  register interrupt register interrupt handling of a specific interrupt.
- unsigned int metal_irq_save_disable (void)
  disable interrupts
- void metal_irq_restore_enable (unsigned int flags)
  restore interrupts to their previous state
- void metal_irq_enable (unsigned int vector)
  metal_irq_enable
- void metal_irq_disable (unsigned int vector)
  metal_irq_disable
- void metal_irq_isr (unsigned int vector)
  default interrupt handler
- int metal_irq_init (void)
- void metal_irq_deinit (void)

Variables

- static struct metal_irqs_state _irqs

8.68.1 Macro Definition Documentation

8.68.1.1 #define MAX_HDS 10

maximum number of handlers per IRQ

8.68.2 Function Documentation

8.68.2.1 void metal_irq_deinit (void)

8.68.2.2 int metal_irq_init (void)

8.68.2.3 void metal_irq_isr (unsigned int *vector*)

default interrupt handler

Parameters

in	*interrupt*	vector

< irq handler

8.68.3 Variable Documentation

8.68.3.1 struct metal_irqs_state _irqs [static]

8.69 lib/system/generic/irq.c File Reference

```
#include <errno.h>
#include "metal/irq.h"
#include "metal/sys.h"
#include "metal/log.h"
#include "metal/mutex.h"
```

Data Structures

- struct metal_irq_hddesc
- struct metal_irqs_state

Macros

- #define MAX_HDS 10

Functions

- int metal_irq_register (int irq, metal_irq_handler hd, struct metal_device *dev, void *drv_id)

  register interrupt register interrupt handling of a specific interrupt.
- unsigned int metal_irq_save_disable (void)

  disable interrupts
- void metal_irq_restore_enable (unsigned int flags)

  restore interrupts to their previous state
- void metal_irq_enable (unsigned int vector)

  metal_irq_enable
- void metal_irq_disable (unsigned int vector)

  metal_irq_disable
- void metal_irq_isr (unsigned int vector)

  default handler
- int metal_irq_init (void)
- void metal_irq_deinit (void)

Variables

- static struct metal_irqs_state _irqs

8.69.1 Macro Definition Documentation

8.69.1.1 #define MAX_HDS 10

maximum number of handlers per IRQ

8.69.2 Function Documentation

8.69.2.1 void metal_irq_deinit (void)

8.69.2.2 int metal_irq_init (void)

8.69.2.3 void metal_irq_isr (unsigned int *vector*)

default handler

default interrupt handler < irq handler

8.69.3 Variable Documentation

8.69.3.1 struct metal_irqs_state _irqs [static]

8.70 lib/system/linux/irq.c File Reference

Linux libmetal irq definitions.

```
#include <pthread.h>
#include <sched.h>
#include "metal/device.h"
#include "metal/irq.h"
#include "metal/sys.h"
#include "metal/mutex.h"
#include <sys/time.h>
#include <sys/eventfd.h>
#include <stdint.h>
#include <errno.h>
#include <unistd.h>
#include <string.h>
#include <poll.h>
```

Data Structures

- struct metal_irq_hddesc
- struct metal_irqs_state

Macros

- #define MAX_IRQS FD_SETSIZE
- #define MAX_HDS 20
- #define METAL_IRQ_STOP 0xFFFFFFFF

Functions

- int metal_irq_register (int irq, metal_irq_handler hd, struct metal_device *dev, void *drv_id)

  register interrupt register interrupt handling of a specific interrupt.
- unsigned int metal_irq_save_disable ()

  disable interrupts
- void metal_irq_restore_enable (unsigned flags)
- void metal_irq_enable (unsigned int vector)

  metal_irq_enable
- void metal_irq_disable (unsigned int vector)

  metal_irq_disable
- static void * metal_linux_irq_handling (void *args)

  IRQ handler.
- int metal_linux_irq_init ()

  irq handling initialization
- void metal_linux_irq_shutdown ()

  irq handling shutdown

Variables

- struct metal_irqs_state _irqs

8.70.1 Detailed Description

Linux libmetal irq definitions.

8.70.2 Macro Definition Documentation

8.70.2.1 #define MAX_HDS 20

maximum number of handlers per IRQ

8.70.2.2 #define MAX_IRQS FD_SETSIZE

maximum number of irqs

8.70.2.3 #define METAL_IRQ_STOP 0xFFFFFFFF

stop interrupts handling thread

8.70.3 Function Documentation

8.70.3.1 void metal_irq_restore_enable (unsigned *flags*)

8.70.3.2 static void* metal_linux_irq_handling (void * *args*) [static]

IRQ handler.

Parameters

in	*args*	not used. required for pthread.

< irq handler descriptro

< irq handler

< metal device which a IRQ belongs to

< A flag to indicate if irq is handled

8.70.3.3 int metal_linux_irq_init ( )

irq handling initialization

Returns

0 on sucess, non-zero on failure

8.70.3.4 void metal_linux_irq_shutdown ( )

irq handling shutdown

8.70.4 Variable Documentation

8.70.4.1 struct metal_irqs_state _irqs

8.71 lib/system/freertos/sleep.c File Reference

```
#include <FreeRTOS.h>
#include <task.h>
#include "metal/sleep.h"
```

Functions

- int metal_sleep_usec (unsigned int usec)

  delay in microseconds delay the next execution in the calling thread fo usec microseconds.

8.72 lib/system/generic/sleep.c File Reference

```
#include "metal/sleep.h"
```

Functions

- int metal_sleep_usec (unsigned int usec)

  delay in microseconds delay the next execution in the calling thread fo usec microseconds.

8.73 lib/system/linux/sleep.c File Reference

```
#include <unistd.h>
#include "metal/sleep.h"
```

Functions

- int metal_sleep_usec (unsigned int usec)

  delay in microseconds delay the next execution in the calling thread fo usec microseconds.

8.74 lib/system/freertos/time.c File Reference

```
#include <FreeRTOS.h>
#include <task.h>
#include "metal/time.h"
```

Functions

- unsigned long long metal_get_timestamp (void)

  get timestamp This function returns the timestampe as unsigned long long value.

8.75 lib/system/generic/time.c File Reference

```
#include "metal/time.h"
```

Functions

- unsigned long long metal_get_timestamp (void)

  get timestamp This function returns the timestampe as unsigned long long value.

8.76 lib/system/linux/time.c File Reference

```
#include <unistd.h>
#include <time.h>
#include "metal/time.h"
```

Macros

- #define NS_PER_S (1000 ∗ 1000 ∗ 1000)

Functions

- unsigned long long metal_get_timestamp (void)

  get timestamp This function returns the timestampe as unsigned long long value.

8.76.1 Macro Definition Documentation

8.76.1.1 #define NS_PER_S (1000 ∗ 1000 ∗ 1000)

8.77 lib/system/freertos/zynq7/sys.c File Reference

```
#include <stdint.h>
#include "xil_cache.h"
#include "xil_mmu.h"
#include "metal/io.h"
#include "xscugic.h"
#include "xil_exception.h"
#include "metal/sys.h"
```

Macros

- #define ARM_AR_MEM_TTB_SIZE 16∗1024
- #define ARM_AR_MEM_TTB_SECT_SIZE 1024∗1024
- #define ARM_AR_MEM_TTB_SECT_SIZE_MASK (∼(ARM_AR_MEM_TTB_SECT_SIZE-1UL))
- #define ARM_AR_MEM_TTB_SECT_TO_DESC_SHIFT (20-2)
- #define ESAL_GE_MEM_32BIT_SET(bit_num) (1UL<<(bit_num))
- #define ARM_AR_MEM_TTB_DESC_BACKWARDS ESAL_GE_MEM_32BIT_SET(4)
- #define ARM_AR_MEM_TTB_DESC_AP_MANAGER

- #define ARM_AR_MEM_TTB_DESC_SECT ESAL_GE_MEM_32BIT_SET(1)
- #define ARM_AR_MEM_TTB_DESC_B ESAL_GE_MEM_32BIT_SET(2)
- #define ARM_AR_MEM_TTB_DESC_C ESAL_GE_MEM_32BIT_SET(3)
- #define ARM_AR_MEM_TTB_DESC_TEX ESAL_GE_MEM_32BIT_SET(12)
- #define ARM_AR_MEM_TTB_DESC_S ESAL_GE_MEM_32BIT_SET(16)
- #define ARM_AR_MEM_TTB_DESC_ALL_ACCESS

Functions

- void sys_irq_restore_enable (void)
- void sys_irq_save_disable (void)
- void metal_machine_cache_flush (void *addr, unsigned int len)
- void metal_machine_cache_invalidate (void *addr, unsigned int len)
- void __attribute__ ((weak))

  poll function until some event happens
- void metal_machine_io_mem_map (metal_phys_addr_t pa, size_t size, unsigned int flags)

Variables

- static unsigned int int_old_val = 0

8.77.1 Macro Definition Documentation

8.77.1.1 #define ARM_AR_MEM_TTB_DESC_ALL_ACCESS

Value:

```
(ARM_AR_MEM_TTB_DESC_AP_MANAGER |           \
                                                ARM_AR_MEM_TTB_DESC_SECT
      )
```

8.77.1.2 #define ARM_AR_MEM_TTB_DESC_AP_MANAGER

Value:

```
(ESAL_GE_MEM_32BIT_SET(10)        |           \
                                                ESAL_GE_MEM_32BIT_SET
      (11))
```

8.77.1.3 #define ARM_AR_MEM_TTB_DESC_B ESAL_GE_MEM_32BIT_SET(2)

8.77.1.4 #define ARM_AR_MEM_TTB_DESC_BACKWARDS ESAL_GE_MEM_32BIT_SET(4)

8.77.1.5 #define ARM_AR_MEM_TTB_DESC_C ESAL_GE_MEM_32BIT_SET(3)

8.77.1.6 #define ARM_AR_MEM_TTB_DESC_S ESAL_GE_MEM_32BIT_SET(16)

8.77.1.7 #define ARM_AR_MEM_TTB_DESC_SECT ESAL_GE_MEM_32BIT_SET(1)

8.77.1.8 #define ARM_AR_MEM_TTB_DESC_TEX ESAL_GE_MEM_32BIT_SET(12)

8.77.1.9 #define ARM_AR_MEM_TTB_SECT_SIZE 1024*1024

8.77.1.10 #define ARM_AR_MEM_TTB_SECT_SIZE_MASK (~(ARM_AR_MEM_TTB_SECT_SIZE-1UL))

8.77.1.11 #define ARM_AR_MEM_TTB_SECT_TO_DESC_SHIFT (20-2)

8.77.1.12 #define ARM_AR_MEM_TTB_SIZE 16*1024

8.77.1.13 #define ESAL_GE_MEM_32BIT_SET(*bit_num*) (1UL<<(bit_num))

8.77.2 Function Documentation

8.77.2.1 void __attribute__ ((weak))

poll function until some event happens

8.77.2.2 void metal_machine_cache_flush (void * *addr,* unsigned int *len*)

8.77.2.3 void metal_machine_cache_invalidate (void * *addr,* unsigned int *len*)

8.77.2.4 void metal_machine_io_mem_map (metal_phys_addr_t *pa,* size_t *size,* unsigned int *flags*)

8.77.2.5 void sys_irq_restore_enable (void)

8.77.2.6 void sys_irq_save_disable (void)

8.77.3 Variable Documentation

8.77.3.1 unsigned int int_old_val = 0 [static]

8.78 lib/system/freertos/zynqmp_r5/sys.c File Reference

```
#include <stdint.h>
#include "xil_cache.h"
#include "xreg_cortexr5.h"
#include "xil_mmu.h"
#include "xil_mpu.h"
#include "xscugic.h"
#include "xil_exception.h"
#include "metal/io.h"
#include "metal/sys.h"
```

Macros

- #define MPU_REGION_SIZE_MIN 0x20

Functions

- void sys_irq_restore_enable (void)
- void sys_irq_save_disable (void)
- void metal_machine_cache_flush (void *addr, unsigned int len)
- void metal_machine_cache_invalidate (void *addr, unsigned int len)
- void __attribute__ ((weak))

  poll function until some event happens
- void metal_machine_io_mem_map (metal_phys_addr_t pa, size_t size, unsigned int flags)

Variables

- static unsigned int int_old_val = 0

8.78.1 Macro Definition Documentation

8.78.1.1 #define MPU_REGION_SIZE_MIN 0x20

8.78.2 Function Documentation

8.78.2.1 void __attribute__ ((weak))

poll function until some event happens

8.78.2.2 void metal_machine_cache_flush (void * *addr,* unsigned int *len*)

8.78.2.3 void metal_machine_cache_invalidate (void * *addr,* unsigned int *len*)

8.78.2.4 void metal_machine_io_mem_map (metal_phys_addr_t *pa,* size_t *size,* unsigned int *flags*)

8.78.2.5 void sys_irq_restore_enable (void)

8.78.2.6 void sys_irq_save_disable (void)

8.78.3 Variable Documentation

8.78.3.1 unsigned int int_old_val = 0 [static]

8.79 lib/system/generic/zynq7/sys.c File Reference

```
#include <stdint.h>
#include "xil_cache.h"
#include "xil_mmu.h"
#include "metal/io.h"
#include "xscugic.h"
#include "xil_exception.h"
#include "metal/sys.h"
```

Macros

- #define ARM_AR_MEM_TTB_SIZE 16*1024
- #define ARM_AR_MEM_TTB_SECT_SIZE 1024*1024
- #define ARM_AR_MEM_TTB_SECT_SIZE_MASK (~(ARM_AR_MEM_TTB_SECT_SIZE-1UL))
- #define ARM_AR_MEM_TTB_SECT_TO_DESC_SHIFT (20-2)
- #define ESAL_GE_MEM_32BIT_SET(bit_num) (1UL<<(bit_num))
- #define ARM_AR_MEM_TTB_DESC_BACKWARDS ESAL_GE_MEM_32BIT_SET(4)
- #define ARM_AR_MEM_TTB_DESC_AP_MANAGER
- #define ARM_AR_MEM_TTB_DESC_SECT ESAL_GE_MEM_32BIT_SET(1)
- #define ARM_AR_MEM_TTB_DESC_B ESAL_GE_MEM_32BIT_SET(2)
- #define ARM_AR_MEM_TTB_DESC_C ESAL_GE_MEM_32BIT_SET(3)
- #define ARM_AR_MEM_TTB_DESC_TEX ESAL_GE_MEM_32BIT_SET(12)
- #define ARM_AR_MEM_TTB_DESC_S ESAL_GE_MEM_32BIT_SET(16)
- #define ARM_AR_MEM_TTB_DESC_ALL_ACCESS

Functions

- void sys_irq_restore_enable (void)
- void sys_irq_save_disable (void)
- void metal_machine_cache_flush (void *addr, unsigned int len)
- void metal_machine_cache_invalidate (void *addr, unsigned int len)
- void __attribute__ ((weak))

  poll function until some event happens
- void metal_machine_io_mem_map (metal_phys_addr_t pa, size_t size, unsigned int flags)

Variables

- static unsigned int int_old_val = XIL_EXCEPTION_ALL

8.79.1 Macro Definition Documentation

8.79.1.1 #define ARM_AR_MEM_TTB_DESC_ALL_ACCESS

Value:

```
(ARM_AR_MEM_TTB_DESC_AP_MANAGER |           \
                                                ARM_AR_MEM_TTB_DESC_SECT
      )
```

8.79.1.2 #define ARM_AR_MEM_TTB_DESC_AP_MANAGER

Value:

```
(ESAL_GE_MEM_32BIT_SET(10)         |           \
                                                ESAL_GE_MEM_32BIT_SET
      (11))
```

8.79.1.3 #define ARM_AR_MEM_TTB_DESC_B ESAL_GE_MEM_32BIT_SET(2)

8.79.1.4 #define ARM_AR_MEM_TTB_DESC_BACKWARDS ESAL_GE_MEM_32BIT_SET(4)

8.79.1.5 #define ARM_AR_MEM_TTB_DESC_C ESAL_GE_MEM_32BIT_SET(3)

8.79.1.6 #define ARM_AR_MEM_TTB_DESC_S ESAL_GE_MEM_32BIT_SET(16)

8.79.1.7 #define ARM_AR_MEM_TTB_DESC_SECT ESAL_GE_MEM_32BIT_SET(1)

8.79.1.8 #define ARM_AR_MEM_TTB_DESC_TEX ESAL_GE_MEM_32BIT_SET(12)

8.79.1.9 #define ARM_AR_MEM_TTB_SECT_SIZE 1024*1024

8.79.1.10 #define ARM_AR_MEM_TTB_SECT_SIZE_MASK (~(ARM_AR_MEM_TTB_SECT_SIZE-1UL))

8.79.1.11 #define ARM_AR_MEM_TTB_SECT_TO_DESC_SHIFT (20-2)

8.79.1.12 #define ARM_AR_MEM_TTB_SIZE 16*1024

8.79.1.13 #define ESAL_GE_MEM_32BIT_SET(*bit_num*) (1UL<<(bit_num))

8.79.2 Function Documentation

8.79.2.1 void __attribute__ ((weak))

poll function until some event happens

8.79.2.2 void metal_machine_cache_flush (void * *addr,* unsigned int *len*)

8.79.2.3 void metal_machine_cache_invalidate (void * *addr,* unsigned int *len*)

8.79.2.4 void metal_machine_io_mem_map (metal_phys_addr_t *pa,* size_t *size,* unsigned int *flags*)

8.79.2.5 void sys_irq_restore_enable (void)

8.79.2.6 void sys_irq_save_disable (void)

8.79.3 Variable Documentation

8.79.3.1 unsigned int int_old_val = XIL_EXCEPTION_ALL [static]

8.80 lib/system/generic/zynqmp_r5/sys.c File Reference

```
#include <stdint.h>
#include "xil_cache.h"
#include "xreg_cortexr5.h"
#include "xil_mmu.h"
#include "xil_mpu.h"
#include "xscugic.h"
#include "xil_exception.h"
#include "metal/io.h"
#include "metal/sys.h"
```

Macros

- #define MPU_REGION_SIZE_MIN 0x20

Functions

- void sys_irq_restore_enable (void)
- void sys_irq_save_disable (void)
- void metal_machine_cache_flush (void *addr, unsigned int len)
- void metal_machine_cache_invalidate (void *addr, unsigned int len)
- void __attribute__ ((weak))

  poll function until some event happens
- void metal_machine_io_mem_map (metal_phys_addr_t pa, size_t size, unsigned int flags)

Variables

- static unsigned int int_old_val = XIL_EXCEPTION_ALL

8.80.1 Macro Definition Documentation

8.80.1.1 #define MPU_REGION_SIZE_MIN 0x20

8.80.2 Function Documentation

8.80.2.1 void __attribute__ ((weak))

poll function until some event happens

8.80.2.2 void metal_machine_cache_flush (void * *addr,* unsigned int *len*)

8.80.2.3 void metal_machine_cache_invalidate (void * *addr,* unsigned int *len*)

8.80.2.4 void metal_machine_io_mem_map (metal_phys_addr_t *pa,* size_t *size,* unsigned int *flags*)

8.80.2.5 void sys_irq_restore_enable (void)

8.80.2.6 void sys_irq_save_disable (void)

8.80.3 Variable Documentation

8.80.3.1 unsigned int int_old_val = XIL_EXCEPTION_ALL [static]

8.81 lib/system/linux/utilities.c File Reference

```
#include "metal/utilities.h"
#include "metal/sys.h"
```

Functions

- int metal_open (const char *path, int shm)
  Open (or create) a file.
- int metal_open_unlinked (const char *path, int shm)
  Open (or create) and unlink a file.
- void metal_randomize_string (char *template)
  Randomize a string.
- void metal_mktemp_template (char template[PATH_MAX], const char *name)
  Create a file name template suitable for use with metal_mktemp.
- int metal_mktemp (char *template, int fifo)
  Create a temporary file or fifo.
- int metal_mktemp_unlinked (char *template)
  Open (or create) and unlink a temporary file.
- int metal_map (int fd, off_t offset, size_t size, int expand, int flags, void **result)
  Map a segment of a file/device.
- int metal_unmap (void *mem, size_t size)
  Unmap a segment of the process address space.
- int metal_mlock (void *mem, size_t size)
  Lock in a region of the process address space.
- int metal_virt2phys (void *addr, unsigned long *phys)

8.81.1 Function Documentation

8.81.1.1 int metal_map (int *fd,* off_t *offset,* size_t *size,* int *expand,* int *flags,* void ** *result*)

Map a segment of a file/device.

This function maps a segment of a file or device into the process address space, after optionally expanding the file if necessary. If required, the file is expanded to hold the requested map area. This is done under and advisory lock, and therefore the called must not have an advisory lock on the file being mmapped.

Parameters

in	*fd*	File descriptor to map.
in	*offset*	Offset in file to map.
in	*size*	Size of region to map.
in	*expand*	Allow file expansion via ftruncate if non-zero.
in	*flags*	Flags for mmap(), MAP_SHARED included implicitly.
out	*result*	Returned pointer to new memory map.

Returns

0 on success, or -errno on error.

8.81.1.2 int metal_mktemp (char * *template,* int *fifo*)

Create a temporary file or fifo.

This function creates a temporary file or fifo, and sets the O_CLOEXEC flag on it.

Parameters

in	*template*	File name template (the last 6 characters must be XXXXXX per mkstemp requirements).
in	*fifo*	Non-zero to create a FIFO instead of a regular file.

Returns

File descriptor.

8.81.1.3 void metal_mktemp_template (char *template[PATH_MAX],* const char * *name*)

Create a file name template suitable for use with metal_mktemp.

Parameters

out	*template*	Template string to be filled in.
in	*name*	Module/user identifier used as part of the generated template.

8.81.1.4 int metal_mktemp_unlinked (char * *template*)

Open (or create) and unlink a temporary file.

This function creates a temporary file, and sets the O_CLOEXEC flag on it. O_CLOEXEC flag set. This file is then unlinked to ensure that it is removed when the last reference to the file is dropped. This ensures that libmetal shared maps are released appropriately once all users of the shared data exit.

Parameters

in	*template*	File name template (the last 6 characters must be XXXXXX per mkstemp requirements).

Returns

File descriptor.

8.81.1.5 int metal_mlock (void * *mem*, size_t *size*)

Lock in a region of the process address space.

Parameters

in	*mem*	Pointer to start of region.
in	*size*	Size of region.

Returns

0 on success, or -errno on error.

8.81.1.6 int metal_open (const char * *path*, int *shm*)

Open (or create) a file.

This function opens or creates a file with read/write permissions and the O_CLOEXEC flag set.

Parameters

in	*path*	File path to open.
in	*shm*	Open shared memory (via shm_open) if non-zero.

Returns

File descriptor.

8.81.1.7 int metal_open_unlinked (const char * *path*, int *shm*)

Open (or create) and unlink a file.

This function opens or creates a file with read/write permissions and the O_CLOEXEC flag set. This file is then unlinked to ensure that it is removed when the last reference to the file is dropped. This ensures that libmetal shared maps are released appropriately once all users of the shared data exit.

Parameters

in	*path*	File path to open.
in	*shm*	Open shared memory (via shm_open) if non-zero.

Returns

File descriptor.

8.81.1.8 void metal_randomize_string (char * *template*)

Randomize a string.

This function randomizes the contents of a string.

Parameters

in	*template*	String to be randomized.

8.81.1.9 int metal_unmap (void * *mem*, size_t *size*)

Unmap a segment of the process address space.

This function unmaps a segment of the process address space.

Parameters

in	*mem*	Segment to unmap.
in	*size*	Size of region to unmap.

Returns

0 on success, or -errno on error.

8.81.1.10 int metal_virt2phys (void * *addr*, unsigned long * *phys*)

8.82 lib/time.h File Reference

```
#include <stdint.h>
#include "metal/sys.h"
```

Functions

- unsigned long long metal_get_timestamp (void)

  get timestamp This function returns the timestampe as unsigned long long value.

8.83 lib/utilities.h File Reference

```
#include <assert.h>
#include <stdint.h>
```

Macros

- #define metal_unused(x) do { (x) = (x); } while (0)
- #define metal_dim(x) (sizeof(x) / sizeof(x[0]))
- #define metal_min(x, y) ((x) < (y) ? (x) : (y))
- #define metal_max(x, y) ((x) > (y) ? (x) : (y))
- #define metal_sign(x) ((x) < 0 ? -1 : ((x) > 0 ? 1 : 0))
- #define metal_align_down(size, align) ((size) & ~((align) - 1))
- #define metal_align_up(size, align) metal_align_down((size) + (align) - 1, align)
- #define metal_div_round_down(num, den) ((num) / (den))
- #define metal_div_round_up(num, den) metal_div_round_down((num) + (den) - 1, (den))
- #define metal_ptr_align_down(ptr, align) (void *)(metal_align_down((uintptr_t)(ptr), (uintptr_t)(align)))
- #define metal_ptr_align_up(ptr, align) (void *)(metal_align_up((uintptr_t)(ptr), (uintptr_t)(align)))
- #define metal_offset_of(structure, member) ((uintptr_t) &(((structure *) 0)->member))

- #define metal_container_of(ptr, structure, member) (void *)((uintptr_t)(ptr) - metal_offset_of(structure, member))
- #define METAL_BITS_PER_ULONG (8 * sizeof(unsigned long))
- #define metal_bit(bit) (1UL << (bit))
- #define metal_bitmap_longs(x) metal_div_round_up((x), METAL_BITS_PER_ULONG)
- #define metal_bitmap_for_each_set_bit(bitmap, bit, max)
- #define metal_bitmap_for_each_clear_bit(bitmap, bit, max)

Functions

- static void metal_bitmap_set_bit (unsigned long *bitmap, int bit)
- static int metal_bitmap_is_bit_set (unsigned long *bitmap, int bit)
- static void metal_bitmap_clear_bit (unsigned long *bitmap, int bit)
- static int metal_bitmap_is_bit_clear (unsigned long *bitmap, int bit)
- static unsigned int metal_bitmap_next_set_bit (unsigned long *bitmap, unsigned int start, unsigned int max)
- static unsigned int metal_bitmap_next_clear_bit (unsigned long *bitmap, unsigned int start, unsigned int max)
- static unsigned long metal_log2 (unsigned long in)

8.84 lib/version.c File Reference

```
#include "metal/config.h"
```

Functions

- int metal_ver_major (void)

  Library major version number.
- int metal_ver_minor (void)

  Library minor version number.
- int metal_ver_patch (void)

  Library patch level.
- const char * metal_ver (void)

  Library version string.

8.85 lib/version.h File Reference

Functions

- int metal_ver_major (void)

  Library major version number.
- int metal_ver_minor (void)

  Library minor version number.
- int metal_ver_patch (void)

  Library patch level.
- const char * metal_ver (void)

  Library version string.

8.86 LICENSE.md File Reference

8.87 README.md File Reference

Index

__attribute__
    freertos/zynq7/sys.c, 109
    freertos/zynqmp_r5/sys.c, 110
    generic/zynq7/sys.c, 112
    generic/zynqmp_r5/sys.c, 113
__metal_mutex_cmpxchg
    system/linux/mutex.h, 92
_irqs
    freertos/irq.c, 102
    generic/irq.c, 104
    linux/irq.c, 106
_metal
    Top Level Interfaces, 46

ATOMIC_FLAG_INIT
    compiler/gcc/atomic.h, 70
ATOMIC_VAR_INIT
    compiler/gcc/atomic.h, 71
Allocation Interfaces, 13
    metal_allocate_memory, 13
    metal_free_memory, 13
atomic_char
    compiler/gcc/atomic.h, 71
atomic_compare_exchange_strong
    compiler/gcc/atomic.h, 69
atomic_compare_exchange_strong_explicit
    compiler/gcc/atomic.h, 69
atomic_compare_exchange_weak
    compiler/gcc/atomic.h, 70
atomic_compare_exchange_weak_explicit
    compiler/gcc/atomic.h, 70
atomic_exchange
    compiler/gcc/atomic.h, 70
atomic_exchange_explicit
    compiler/gcc/atomic.h, 70
atomic_fetch_add
    compiler/gcc/atomic.h, 70
atomic_fetch_add_explicit
    compiler/gcc/atomic.h, 70
atomic_fetch_and
    compiler/gcc/atomic.h, 70
atomic_fetch_and_explicit
    compiler/gcc/atomic.h, 70
atomic_fetch_or
    compiler/gcc/atomic.h, 70
atomic_fetch_or_explicit
    compiler/gcc/atomic.h, 70
atomic_fetch_sub
    compiler/gcc/atomic.h, 70
atomic_fetch_sub_explicit
    compiler/gcc/atomic.h, 70
atomic_fetch_xor
    compiler/gcc/atomic.h, 70
atomic_fetch_xor_explicit
    compiler/gcc/atomic.h, 70
atomic_flag
    compiler/gcc/atomic.h, 71
atomic_flag_clear
    compiler/gcc/atomic.h, 70
atomic_flag_clear_explicit
    compiler/gcc/atomic.h, 70
atomic_flag_test_and_set
    compiler/gcc/atomic.h, 70
atomic_flag_test_and_set_explicit
    compiler/gcc/atomic.h, 70
atomic_init
    compiler/gcc/atomic.h, 70
atomic_int
    compiler/gcc/atomic.h, 71
atomic_is_lock_free
    compiler/gcc/atomic.h, 70
atomic_llong
    compiler/gcc/atomic.h, 71
atomic_load
    compiler/gcc/atomic.h, 71
atomic_load_explicit
    compiler/gcc/atomic.h, 71
atomic_long
    compiler/gcc/atomic.h, 71
atomic_short
    compiler/gcc/atomic.h, 71
atomic_signal_fence
    compiler/gcc/atomic.h, 71
atomic_store
    compiler/gcc/atomic.h, 71
atomic_store_explicit
    compiler/gcc/atomic.h, 71
atomic_thread_fence
    compiler/gcc/atomic.h, 71
atomic_uchar
    compiler/gcc/atomic.h, 71
atomic_uint
    compiler/gcc/atomic.h, 71
atomic_ullong
    compiler/gcc/atomic.h, 71
atomic_ulong
    compiler/gcc/atomic.h, 71
atomic_ushort
    compiler/gcc/atomic.h, 71

bus
    linux_bus, 53
    metal_device, 58
Bus Abstraction, 18
    metal_bus_find, 18
    metal_bus_register, 19
    metal_bus_unregister, 19
    metal_device_close, 19
    metal_device_io_region, 19
    metal_device_open, 19
    metal_generic_bus, 20
    metal_register_generic_device, 20
bus_close
    metal_bus_ops, 56
bus_list
    metal_common_state, 57
bus_name
    linux_bus, 53

CACHE Interfaces, 14
    metal_cache_flush, 14
    metal_cache_invalidate, 14
close
    IO Interfaces, 31
cls_name
    linux_driver, 54
cls_path
    linux_device, 54
common
    metal_state, 65
compiler/gcc/atomic.h
    memory_order_acq_rel, 71
    memory_order_acquire, 71
    memory_order_consume, 71
    memory_order_relaxed, 71
    memory_order_release, 71
    memory_order_seq_cst, 71
compiler/gcc/atomic.h
    ATOMIC_FLAG_INIT, 70
    ATOMIC_VAR_INIT, 71
    atomic_char, 71
    atomic_compare_exchange_strong, 69
    atomic_compare_exchange_strong_explicit, 69
    atomic_compare_exchange_weak, 70
    atomic_compare_exchange_weak_explicit, 70
    atomic_exchange, 70
    atomic_exchange_explicit, 70
    atomic_fetch_add, 70
    atomic_fetch_add_explicit, 70
    atomic_fetch_and, 70
    atomic_fetch_and_explicit, 70
    atomic_fetch_or, 70
    atomic_fetch_or_explicit, 70
    atomic_fetch_sub, 70
    atomic_fetch_sub_explicit, 70
    atomic_fetch_xor, 70
    atomic_fetch_xor_explicit, 70
    atomic_flag, 71
    atomic_flag_clear, 70
    atomic_flag_clear_explicit, 70
    atomic_flag_test_and_set, 70
    atomic_flag_test_and_set_explicit, 70
    atomic_init, 70
    atomic_int, 71
    atomic_is_lock_free, 70
    atomic_llong, 71
    atomic_load, 71
    atomic_load_explicit, 71
    atomic_long, 71
    atomic_short, 71
    atomic_signal_fence, 71
    atomic_store, 71
    atomic_store_explicit, 71
    atomic_thread_fence, 71
    atomic_uchar, 71
    atomic_uint, 71
    atomic_ullong, 71
    atomic_ulong, 71
    atomic_ushort, 71
    memory_order, 71
compiler/gcc/compiler.h
    metal_align, 72
Condition Variable Interfaces, 15
    metal_condition_broadcast, 15
    metal_condition_init, 15
    metal_condition_signal, 15
    metal_condition_wait, 15
config.h
    HAVE_FUTEX_H, 75
    HAVE_STDATOMIC_H, 75
    METAL_MACHINE, 75
    METAL_MACHINE_, 75
    METAL_PROCESSOR, 75
    METAL_PROCESSOR_, 75
    METAL_SYSTEM, 75
    METAL_SYSTEM_, 75
    METAL_VER, 75
    METAL_VER_MAJOR, 75
    METAL_VER_MINOR, 75
    METAL_VER_PATCH, 76

DMA Interfaces, 21
    METAL_DMA_DEV_R, 21
    METAL_DMA_DEV_W, 21
    METAL_DMA_DEV_WR, 21
    metal_dma_map, 21
    metal_dma_unmap, 22
data_fd
    metal_state, 65
dev
    metal_irq_hddesc, 61
dev_close
    linux_driver, 54
    metal_bus_ops, 56
dev_dma_map
    linux_driver, 54
    metal_bus_ops, 56
dev_dma_unmap

linux_driver, 54
metal_bus_ops, 56
dev_irq_ack
linux_driver, 54
metal_bus_ops, 56
dev_name
linux_device, 54
dev_open
linux_driver, 54
metal_bus_ops, 56
dev_path
linux_device, 54
device
linux_device, 54
device.c
metal_generic_dev_dma_map, 77
metal_generic_dev_dma_unmap, 77
metal_generic_dev_open, 77
devices
metal_bus, 55
drivers
linux_bus, 53
drv_id
metal_irq_hddesc, 61
drv_name
linux_driver, 55

fd
linux_device, 54
for_each_linux_bus
system/linux/device.c, 79
for_each_linux_driver
system/linux/device.c, 79
freertos/cache.c
metal_machine_cache_flush, 100
metal_machine_cache_invalidate, 100
freertos/irq.c
_irqs, 102
MAX_HDS, 102
metal_irq_deinit, 102
metal_irq_init, 102
metal_irq_isr, 102
freertos/zynq7/sys.c
__attribute__, 109
int_old_val, 109
metal_machine_cache_flush, 109
metal_machine_cache_invalidate, 109
metal_machine_io_mem_map, 109
sys_irq_restore_enable, 109
sys_irq_save_disable, 109
freertos/zynqmp_r5/sys.c
__attribute__, 110
int_old_val, 110
metal_machine_cache_flush, 110
metal_machine_cache_invalidate, 110
metal_machine_io_mem_map, 110
sys_irq_restore_enable, 110
sys_irq_save_disable, 110
generic/cache.c
metal_machine_cache_flush, 100
metal_machine_cache_invalidate, 100
generic/condition.c
metal_generic_default_poll, 101
generic/irq.c
_irqs, 104
MAX_HDS, 103
metal_irq_deinit, 103
metal_irq_init, 103
metal_irq_isr, 103
generic/zynq7/sys.c
__attribute__, 112
int_old_val, 112
metal_machine_cache_flush, 112
metal_machine_cache_invalidate, 112
metal_machine_io_mem_map, 112
sys_irq_restore_enable, 112
sys_irq_save_disable, 112
generic/zynqmp_r5/sys.c
__attribute__, 113
int_old_val, 113
metal_machine_cache_flush, 113
metal_machine_cache_invalidate, 113
metal_machine_io_mem_map, 113
sys_irq_restore_enable, 113
sys_irq_save_disable, 113
generic_device_list
metal_common_state, 57
generic_shmem_list
metal_common_state, 57

HAVE_FUTEX_H
config.h, 75
HAVE_STDATOMIC_H
config.h, 75
hd
metal_irq_hddesc, 61
hds
metal_irqs_state, 61

IO Interfaces, 23
close, 31
METAL_CACHE_UNKNOWN, 24
METAL_CACHE_WB, 24
METAL_CACHE_WT, 24
METAL_IO_MAPPED, 24
METAL_MEM_MAPPED, 25
METAL_SHARED_MEM, 25
METAL_UNCACHED, 25
mem_flags, 31
metal_io_finish, 25
metal_io_init, 25
metal_io_mem_map, 27
metal_io_phys, 27
metal_io_phys_to_offset, 27
metal_io_phys_to_virt, 27
metal_io_read, 28
metal_io_read16, 25

metal_io_read16_explicit, 25
metal_io_read32, 25
metal_io_read32_explicit, 25
metal_io_read64, 25
metal_io_read64_explicit, 25
metal_io_read8, 25
metal_io_read8_explicit, 25
metal_io_region_size, 28
metal_io_virt, 28
metal_io_virt_to_offset, 28
metal_io_virt_to_phys, 29
metal_io_write, 29
metal_io_write16, 25
metal_io_write16_explicit, 25
metal_io_write32, 25
metal_io_write32_explicit, 25
metal_io_write64, 25
metal_io_write64_explicit, 25
metal_io_write8, 25
metal_io_write8_explicit, 25
metal_memcpy_io, 29
metal_memset_io, 29
ops, 31
page_mask, 31
page_shift, 31
physmap, 31
read, 31
size, 31
virt, 31
write, 31
int_old_val
freertos/zynq7/sys.c, 109
freertos/zynqmp_r5/sys.c, 110
generic/zynq7/sys.c, 112
generic/zynqmp_r5/sys.c, 113
Interrupt Handling Interfaces, 32
METAL_IRQ_HANDLED, 32
metal_irq_disable, 33
metal_irq_enable, 33
metal_irq_handler, 32
metal_irq_register, 33
metal_irq_restore_enable, 33
metal_irq_save_disable, 33
intr_enable
metal_irqs_state, 61
io
metal_generic_shmem, 59
metal_sg, 64
metal_shmem, 64
io.c
metal_generic_memcpy_io, 84
metal_generic_memset_io, 84
irq_info
metal_device, 58
irq_lock
metal_irqs_state, 61
irq_num
metal_device, 58
irq_pthread
metal_irqs_state, 61
irq_reg_fd
metal_irqs_state, 61
irq_reg_stat
metal_irqs_state, 61
irq_state
metal_irqs_state, 62

LOG_ALERT
Library Logging Interfaces, 37
LOG_CRITICAL
Library Logging Interfaces, 37
LOG_DEBUG
Library Logging Interfaces, 37
LOG_EMERGENCY
Library Logging Interfaces, 37
LOG_ERROR
Library Logging Interfaces, 37
LOG_INFO
Library Logging Interfaces, 37
LOG_NOTICE
Library Logging Interfaces, 37
LOG_WARNING
Library Logging Interfaces, 37
LICENSE.md, 118
ldrv
linux_device, 54
len
metal_sg, 64
lib/alloc.h, 67
lib/atomic.h, 68
lib/cache.h, 72
lib/compiler.h, 72
lib/compiler/gcc/atomic.h, 68
lib/compiler/gcc/compiler.h, 72
lib/condition.h, 72
lib/config.h, 75
lib/cpu.h, 76
lib/device.c, 77
lib/device.h, 80
lib/dma.c, 81
lib/dma.h, 81
lib/init.c, 82
lib/io.c, 84
lib/io.h, 85
lib/irq.h, 87
lib/list.h, 88
lib/log.c, 89
lib/log.h, 89
lib/mutex.h, 90
lib/processor/aarch64/atomic.h, 71
lib/processor/aarch64/cpu.h, 76
lib/processor/arm/atomic.h, 72
lib/processor/arm/cpu.h, 76
lib/processor/x86_64/atomic.h, 72
lib/processor/x86_64/cpu.h, 76
lib/shmem.c, 93
lib/shmem.h, 94

lib/sleep.h, 94
lib/spinlock.h, 95
lib/sys.h, 95
lib/system/freertos/alloc.h, 67
lib/system/freertos/cache.c, 99
lib/system/freertos/condition.c, 100
lib/system/freertos/condition.h, 72
lib/system/freertos/init.c, 82
lib/system/freertos/io.c, 84
lib/system/freertos/irq.c, 101
lib/system/freertos/irq.h, 87
lib/system/freertos/mutex.h, 90
lib/system/freertos/shmem.c, 93
lib/system/freertos/sleep.c, 106
lib/system/freertos/sys.h, 96
lib/system/freertos/time.c, 106
lib/system/freertos/zynq7/sys.c, 107
lib/system/freertos/zynq7/sys.h, 96
lib/system/freertos/zynqmp_r5/sys.c, 109
lib/system/freertos/zynqmp_r5/sys.h, 97
lib/system/generic/alloc.h, 67
lib/system/generic/cache.c, 100
lib/system/generic/condition.c, 101
lib/system/generic/condition.h, 73
lib/system/generic/init.c, 82
lib/system/generic/io.c, 85
lib/system/generic/irq.c, 102
lib/system/generic/irq.h, 88
lib/system/generic/mutex.h, 91
lib/system/generic/shmem.c, 93
lib/system/generic/sleep.c, 106
lib/system/generic/sys.h, 97
lib/system/generic/time.c, 107
lib/system/generic/zynq7/sys.c, 110
lib/system/generic/zynq7/sys.h, 98
lib/system/generic/zynqmp_r5/sys.c, 112
lib/system/generic/zynqmp_r5/sys.h, 98
lib/system/linux/alloc.h, 68
lib/system/linux/cache.c, 100
lib/system/linux/condition.c, 101
lib/system/linux/condition.h, 74
lib/system/linux/device.c, 78
lib/system/linux/init.c, 83
lib/system/linux/io.c, 85
lib/system/linux/irq.c, 104
lib/system/linux/irq.h, 88
lib/system/linux/mutex.h, 92
lib/system/linux/shmem.c, 93
lib/system/linux/sleep.c, 106
lib/system/linux/sys.h, 99
lib/system/linux/time.c, 107
lib/system/linux/utilities.c, 113
lib/time.h, 116
lib/utilities.h, 116
lib/version.c, 117
lib/version.h, 117
Library Logging Interfaces, 36
  LOG_ALERT, 37
  LOG_CRITICAL, 37
  LOG_DEBUG, 37
  LOG_EMERGENCY, 37
  LOG_ERROR, 37
  LOG_INFO, 37
  LOG_NOTICE, 37
  LOG_WARNING, 37
  metal_default_log_handler, 37
  metal_get_log_handler, 37
  metal_get_log_level, 37
  metal_log, 36
  metal_log_handler, 36
  metal_log_level, 37
  metal_set_log_handler, 37
  metal_set_log_level, 38
Library Version Interfaces, 51
  metal_ver, 51
  metal_ver_major, 51
  metal_ver_minor, 51
  metal_ver_patch, 52
linux/irq.c
  _irqs, 106
  MAX_HDS, 105
  MAX_IRQS, 105
  METAL_IRQ_STOP, 105
  metal_irq_restore_enable, 105
  metal_linux_irq_handling, 105
  metal_linux_irq_init, 105
  metal_linux_irq_shutdown, 105
linux/time.c
  NS_PER_S, 107
linux_bus, 53
  bus, 53
  bus_name, 53
  drivers, 53
  sbus, 53
  system/linux/device.c, 80
linux_device, 53
  cls_path, 54
  dev_name, 54
  dev_path, 54
  device, 54
  fd, 54
  ldrv, 54
  override, 54
  region_phys, 54
  sdev, 54
linux_driver, 54
  cls_name, 54
  dev_close, 54
  dev_dma_map, 54
  dev_dma_unmap, 54
  dev_irq_ack, 54
  dev_open, 54
  drv_name, 55
  mod_name, 55
  sdrv, 55
List Primitives, 34

METAL_DECLARE_LIST, 34
metal_list_add_after, 34
metal_list_add_before, 34
metal_list_add_head, 34
metal_list_add_tail, 34
metal_list_del, 34
metal_list_first, 34
metal_list_for_each, 34
metal_list_init, 34
metal_list_is_empty, 34
log_handler
metal_common_state, 57
metal_init_params, 59
log_level
metal_common_state, 57
metal_init_params, 59

m
metal_condition, 57
metal_mutex_t, 62
MAX_DRIVERS
system/linux/device.c, 79
MAX_HDS
freertos/irq.c, 102
generic/irq.c, 103
linux/irq.c, 105
MAX_IRQS
linux/irq.c, 105
system/freertos/zynq7/sys.h, 96
system/freertos/zynqmp_r5/sys.h, 97
system/generic/zynq7/sys.h, 98
system/generic/zynqmp_r5/sys.h, 98
MAX_PAGE_SIZES
system/linux/sys.h, 99
METAL_BAD_IRQ
Top Level Interfaces, 45
METAL_BAD_OFFSET
Top Level Interfaces, 45
METAL_BAD_PHYS
Top Level Interfaces, 45
METAL_CACHE_UNKNOWN
IO Interfaces, 24
METAL_CACHE_WB
IO Interfaces, 24
METAL_CACHE_WT
IO Interfaces, 24
METAL_DECLARE_LIST
List Primitives, 34
METAL_DMA_DEV_R
DMA Interfaces, 21
METAL_DMA_DEV_W
DMA Interfaces, 21
METAL_DMA_DEV_WR
DMA Interfaces, 21
METAL_INIT_DEFAULTS
Top Level Interfaces, 45
METAL_INVALID_VADDR
system/linux/sys.h, 99
METAL_IO_MAPPED
IO Interfaces, 24
METAL_IRQ_HANDLED
Interrupt Handling Interfaces, 32
METAL_IRQ_STOP
linux/irq.c, 105
METAL_MACHINE
config.h, 75
METAL_MACHINE_
config.h, 75
METAL_MEM_MAPPED
IO Interfaces, 25
METAL_MUTEX_INIT
system/freertos/mutex.h, 91
system/generic/mutex.h, 91
system/linux/mutex.h, 92
METAL_PROCESSOR
config.h, 75
METAL_PROCESSOR_
config.h, 75
METAL_SHARED_MEM
IO Interfaces, 25
METAL_SPINLOCK_INIT
Spinlock Interfaces, 43
METAL_SYSTEM
config.h, 75
METAL_SYSTEM_
config.h, 75
METAL_UNCACHED
IO Interfaces, 25
METAL_VER
config.h, 75
METAL_VER_MAJOR
config.h, 75
METAL_VER_MINOR
config.h, 75
METAL_VER_PATCH
config.h, 76
mem_flags
IO Interfaces, 31
memory_order_acq_rel
compiler/gcc/atomic.h, 71
memory_order_acquire
compiler/gcc/atomic.h, 71
memory_order_consume
compiler/gcc/atomic.h, 71
memory_order_relaxed
compiler/gcc/atomic.h, 71
memory_order_release
compiler/gcc/atomic.h, 71
memory_order_seq_cst
compiler/gcc/atomic.h, 71
memory_order
compiler/gcc/atomic.h, 71
metal_add_page_size
system/linux/init.c, 83
metal_align
compiler/gcc/compiler.h, 72
metal_align_down

Simple Utilities, 48
metal_align_up
Simple Utilities, 48
metal_allocate_memory
Allocation Interfaces, 13
system/freertos/alloc.h, 67
system/generic/alloc.h, 68
system/linux/alloc.h, 68
metal_bit
Simple Utilities, 48
metal_bitmap_clear_bit
Simple Utilities, 50
metal_bitmap_for_each_clear_bit
Simple Utilities, 48
metal_bitmap_for_each_set_bit
Simple Utilities, 48
metal_bitmap_is_bit_clear
Simple Utilities, 50
metal_bitmap_is_bit_set
Simple Utilities, 50
metal_bitmap_longs
Simple Utilities, 49
metal_bitmap_next_clear_bit
Simple Utilities, 50
metal_bitmap_next_set_bit
Simple Utilities, 50
metal_bitmap_set_bit
Simple Utilities, 50
metal_bus, 55
devices, 55
name, 55
node, 55
ops, 55
metal_bus_find
Bus Abstraction, 18
metal_bus_ops, 55
bus_close, 56
dev_close, 56
dev_dma_map, 56
dev_dma_unmap, 56
dev_irq_ack, 56
dev_open, 56
metal_bus_register
Bus Abstraction, 19
metal_bus_unregister
Bus Abstraction, 19
metal_cache_flush
CACHE Interfaces, 14
metal_cache_invalidate
CACHE Interfaces, 14
metal_common_state, 56
bus_list, 57
generic_device_list, 57
generic_shmem_list, 57
log_handler, 57
log_level, 57
metal_condition, 57
m, 57
v, 57
waiters, 57
wakeups, 57
metal_condition_broadcast
Condition Variable Interfaces, 15
system/freertos/condition.h, 73
system/generic/condition.h, 74
system/linux/condition.h, 74
metal_condition_init
Condition Variable Interfaces, 15
system/freertos/condition.h, 73
system/generic/condition.h, 74
system/linux/condition.h, 74
metal_condition_signal
Condition Variable Interfaces, 15
system/freertos/condition.h, 73
system/generic/condition.h, 74
system/linux/condition.h, 75
metal_condition_wait
Condition Variable Interfaces, 15
metal_container_of
Simple Utilities, 49
metal_cpu_yield
processor/aarch64/cpu.h, 76
processor/arm/cpu.h, 76
processor/x86_64/cpu.h, 76
metal_default_log_handler
Library Logging Interfaces, 37
metal_device, 58
bus, 58
irq_info, 58
irq_num, 58
name, 58
node, 58
num_regions, 58
regions, 58
metal_device_close
Bus Abstraction, 19
metal_device_io_region
Bus Abstraction, 19
metal_device_open
Bus Abstraction, 19
metal_dim
Simple Utilities, 49
metal_div_round_down
Simple Utilities, 49
metal_div_round_up
Simple Utilities, 49
metal_dma_map
DMA Interfaces, 21
metal_dma_unmap
DMA Interfaces, 22
metal_finish
Top Level Interfaces, 46
metal_free_memory
Allocation Interfaces, 13
system/freertos/alloc.h, 67
system/generic/alloc.h, 68

system/linux/alloc.h, 68
metal_generic_bus
Bus Abstraction, 20
metal_generic_default_poll
generic/condition.c, 101
metal_generic_dev_dma_map
device.c, 77
metal_generic_dev_dma_unmap
device.c, 77
metal_generic_dev_open
device.c, 77
metal_generic_memcpy_io
io.c, 84
metal_generic_memset_io
io.c, 84
metal_generic_shmem, 58
io, 59
name, 59
node, 59
metal_get_log_handler
Library Logging Interfaces, 37
metal_get_log_level
Library Logging Interfaces, 37
metal_get_timestamp
TIME Interfaces, 47
metal_init
Top Level Interfaces, 46
metal_init_page_sizes
system/linux/init.c, 83
metal_init_params, 59
log_handler, 59
log_level, 59
metal_io_finish
IO Interfaces, 25
metal_io_init
IO Interfaces, 25
metal_io_mem_map
IO Interfaces, 27
metal_io_ops, 60
metal_io_phys
IO Interfaces, 27
metal_io_phys_to_offset
IO Interfaces, 27
metal_io_phys_to_virt
IO Interfaces, 27
metal_io_read
IO Interfaces, 28
metal_io_read16
IO Interfaces, 25
metal_io_read16_explicit
IO Interfaces, 25
metal_io_read32
IO Interfaces, 25
metal_io_read32_explicit
IO Interfaces, 25
metal_io_read64
IO Interfaces, 25
metal_io_read64_explicit
IO Interfaces, 25
metal_io_read8
IO Interfaces, 25
metal_io_read8_explicit
IO Interfaces, 25
metal_io_region, 60
metal_io_region_size
IO Interfaces, 28
metal_io_virt
IO Interfaces, 28
metal_io_virt_to_offset
IO Interfaces, 28
metal_io_virt_to_phys
IO Interfaces, 29
metal_io_write
IO Interfaces, 29
metal_io_write16
IO Interfaces, 25
metal_io_write16_explicit
IO Interfaces, 25
metal_io_write32
IO Interfaces, 25
metal_io_write32_explicit
IO Interfaces, 25
metal_io_write64
IO Interfaces, 25
metal_io_write64_explicit
IO Interfaces, 25
metal_io_write8
IO Interfaces, 25
metal_io_write8_explicit
IO Interfaces, 25
metal_irq_deinit
freertos/irq.c, 102
generic/irq.c, 103
system/freertos/init.c, 82
system/generic/init.c, 83
metal_irq_disable
Interrupt Handling Interfaces, 33
metal_irq_enable
Interrupt Handling Interfaces, 33
metal_irq_handler
Interrupt Handling Interfaces, 32
metal_irq_hddesc, 60
dev, 61
drv_id, 61
hd, 61
metal_irq_init
freertos/irq.c, 102
generic/irq.c, 103
system/freertos/init.c, 82
system/generic/init.c, 83
metal_irq_isr
freertos/irq.c, 102
generic/irq.c, 103
system/freertos/irq.h, 87
system/generic/irq.h, 88
metal_irq_register

Interrupt Handling Interfaces, 33
metal_irq_restore_enable
Interrupt Handling Interfaces, 33
linux/irq.c, 105
metal_irq_save_disable
Interrupt Handling Interfaces, 33
metal_irq_t
Top Level Interfaces, 46
metal_irqs_state, 61
hds, 61
intr_enable, 61
irq_lock, 61
irq_pthread, 61
irq_reg_fd, 61
irq_reg_stat, 61
irq_state, 62
metal_linux_bus_close
system/linux/device.c, 79
metal_linux_bus_finish
system/linux/device.c, 79
metal_linux_bus_init
system/linux/device.c, 79
metal_linux_bus_ops
system/linux/device.c, 80
metal_linux_dev_close
system/linux/device.c, 79
metal_linux_dev_dma_map
system/linux/device.c, 79
metal_linux_dev_dma_unmap
system/linux/device.c, 79
metal_linux_dev_irq_ack
system/linux/device.c, 79
metal_linux_dev_open
system/linux/device.c, 79
metal_linux_irq_handling
linux/irq.c, 105
metal_linux_irq_init
linux/irq.c, 105
system/linux/init.c, 83
metal_linux_irq_shutdown
linux/irq.c, 105
system/linux/init.c, 83
metal_linux_probe_bus
system/linux/device.c, 79
metal_linux_probe_driver
system/linux/device.c, 79
metal_linux_register_bus
system/linux/device.c, 79
metal_list, 62
next, 62
prev, 62
metal_list_add_after
List Primitives, 34
metal_list_add_before
List Primitives, 34
metal_list_add_head
List Primitives, 34
metal_list_add_tail
List Primitives, 34
metal_list_del
List Primitives, 34
metal_list_first
List Primitives, 34
metal_list_for_each
List Primitives, 34
metal_list_init
List Primitives, 34
metal_list_is_empty
List Primitives, 34
metal_log
Library Logging Interfaces, 36
metal_log2
Simple Utilities, 50
metal_log_handler
Library Logging Interfaces, 36
metal_log_level
Library Logging Interfaces, 37
metal_machine_cache_flush
freertos/cache.c, 100
freertos/zynq7/sys.c, 109
freertos/zynqmp_r5/sys.c, 110
generic/cache.c, 100
generic/zynq7/sys.c, 112
generic/zynqmp_r5/sys.c, 113
metal_machine_cache_invalidate
freertos/cache.c, 100
freertos/zynq7/sys.c, 109
freertos/zynqmp_r5/sys.c, 110
generic/cache.c, 100
generic/zynq7/sys.c, 112
generic/zynqmp_r5/sys.c, 113
metal_machine_io_mem_map
freertos/zynq7/sys.c, 109
freertos/zynqmp_r5/sys.c, 110
generic/zynq7/sys.c, 112
generic/zynqmp_r5/sys.c, 113
system/freertos/io.c, 84
system/generic/io.c, 85
metal_map
utilities.c, 114
metal_max
Simple Utilities, 49
metal_memcpy_io
IO Interfaces, 29
metal_memset_io
IO Interfaces, 29
metal_min
Simple Utilities, 49
metal_mktemp
utilities.c, 114
metal_mktemp_template
utilities.c, 114
metal_mktemp_unlinked
utilities.c, 114
metal_mlock
utilities.c, 115

metal_mutex_acquire
Mutex Interfaces, 39
system/freertos/mutex.h, 91
system/generic/mutex.h, 91
system/linux/mutex.h, 92
metal_mutex_deinit
Mutex Interfaces, 39
system/freertos/mutex.h, 91
system/generic/mutex.h, 91
system/linux/mutex.h, 92
metal_mutex_init
Mutex Interfaces, 39
system/freertos/mutex.h, 91
system/generic/mutex.h, 91
system/linux/mutex.h, 92
metal_mutex_is_acquired
Mutex Interfaces, 39
system/freertos/mutex.h, 91
system/generic/mutex.h, 91
system/linux/mutex.h, 92
metal_mutex_release
Mutex Interfaces, 40
system/freertos/mutex.h, 91
system/generic/mutex.h, 92
system/linux/mutex.h, 92
metal_mutex_t, 62
m, 62
v, 62
metal_mutex_try_acquire
Mutex Interfaces, 40
system/freertos/mutex.h, 91
system/generic/mutex.h, 92
system/linux/mutex.h, 92
metal_offset_of
Simple Utilities, 49
metal_open
utilities.c, 115
metal_open_unlinked
utilities.c, 115
metal_page_size, 63
mmap_flags, 63
page_shift, 63
page_size, 63
path, 63
metal_pagesize_compare
system/linux/init.c, 84
metal_phys_addr_t
Top Level Interfaces, 46
metal_ptr_align_down
Simple Utilities, 49
metal_ptr_align_up
Simple Utilities, 49
metal_randomize_string
utilities.c, 115
metal_register_generic_device
Bus Abstraction, 20
metal_set_log_handler
Library Logging Interfaces, 37
metal_set_log_level
Library Logging Interfaces, 38
metal_sg, 63
io, 64
len, 64
virt, 64
metal_shmem, 64
io, 64
phys, 64
metal_shmem_io_close
system/linux/shmem.c, 94
metal_shmem_io_ops
system/linux/shmem.c, 94
metal_shmem_open
Shared Memory Interfaces, 41
metal_shmem_open_generic
shmem.c, 93
metal_shmem_register_generic
Shared Memory Interfaces, 41
metal_shmem_try_map
system/linux/shmem.c, 94
metal_sign
Simple Utilities, 49
metal_sleep_usec
Sleep Interfaces, 42
metal_spinlock, 64
v, 65
metal_spinlock_acquire
Spinlock Interfaces, 43
metal_spinlock_init
Spinlock Interfaces, 43
metal_spinlock_release
Spinlock Interfaces, 43
metal_state, 65
common, 65
data_fd, 65
num_page_sizes, 65
page_shift, 65
page_size, 65
page_sizes, 66
pagemap_fd, 66
sysfs_path, 66
tmp_path, 66
metal_sys_finish
system/freertos/init.c, 82
system/generic/init.c, 83
system/linux/init.c, 84
metal_sys_init
system/freertos/init.c, 82
system/generic/init.c, 83
system/linux/init.c, 84
metal_uio_dev_bind
system/linux/device.c, 79
metal_uio_dev_close
system/linux/device.c, 79
metal_uio_dev_dma_map
system/linux/device.c, 79
metal_uio_dev_dma_unmap

system/linux/device.c, 79
metal_uio_dev_irq_ack
system/linux/device.c, 79
metal_uio_dev_open
system/linux/device.c, 79
metal_uio_read_map_attr
system/linux/device.c, 79
metal_unmap
utilities.c, 116
metal_unused
Simple Utilities, 50
metal_ver
Library Version Interfaces, 51
metal_ver_major
Library Version Interfaces, 51
metal_ver_minor
Library Version Interfaces, 51
metal_ver_patch
Library Version Interfaces, 52
metal_virt2phys
utilities.c, 116
mmap_flags
metal_page_size, 63
mod_name
linux_driver, 55
Mutex Interfaces, 39
metal_mutex_acquire, 39
metal_mutex_deinit, 39
metal_mutex_init, 39
metal_mutex_is_acquired, 39
metal_mutex_release, 40
metal_mutex_try_acquire, 40

NS_PER_S
linux/time.c, 107
name
metal_bus, 55
metal_device, 58
metal_generic_shmem, 59
next
metal_list, 62
node
metal_bus, 55
metal_device, 58
metal_generic_shmem, 59
num_page_sizes
metal_state, 65
num_regions
metal_device, 58

ops
IO Interfaces, 31
metal_bus, 55
override
linux_device, 54

page_mask
IO Interfaces, 31
page_shift
IO Interfaces, 31
metal_page_size, 63
metal_state, 65
page_size
metal_page_size, 63
metal_state, 65
page_sizes
metal_state, 66
pagemap_fd
metal_state, 66
path
metal_page_size, 63
phys
metal_shmem, 64
physmap
IO Interfaces, 31
prev
metal_list, 62
processor/aarch64/cpu.h
metal_cpu_yield, 76
processor/arm/cpu.h
metal_cpu_yield, 76
processor/x86_64/cpu.h
metal_cpu_yield, 76

README.md, 118
read
IO Interfaces, 31
region_phys
linux_device, 54
regions
metal_device, 58

sbus
linux_bus, 53
sdev
linux_device, 54
sdrv
linux_driver, 55
Shared Memory Interfaces, 41
metal_shmem_open, 41
metal_shmem_register_generic, 41
shmem.c
metal_shmem_open_generic, 93
Simple Utilities, 48
metal_align_down, 48
metal_align_up, 48
metal_bit, 48
metal_bitmap_clear_bit, 50
metal_bitmap_for_each_clear_bit, 48
metal_bitmap_for_each_set_bit, 48
metal_bitmap_is_bit_clear, 50
metal_bitmap_is_bit_set, 50
metal_bitmap_longs, 49
metal_bitmap_next_clear_bit, 50
metal_bitmap_next_set_bit, 50
metal_bitmap_set_bit, 50
metal_container_of, 49
metal_dim, 49

metal_div_round_down, 49
metal_div_round_up, 49
metal_log2, 50
metal_max, 49
metal_min, 49
metal_offset_of, 49
metal_ptr_align_down, 49
metal_ptr_align_up, 49
metal_sign, 49
metal_unused, 50
size
IO Interfaces, 31
Sleep Interfaces, 42
metal_sleep_usec, 42
Spinlock Interfaces, 43
METAL_SPINLOCK_INIT, 43
metal_spinlock_acquire, 43
metal_spinlock_init, 43
metal_spinlock_release, 43
sys_irq_disable
system/freertos/zynq7/sys.h, 97
system/freertos/zynqmp_r5/sys.h, 97
system/generic/zynq7/sys.h, 98
system/generic/zynqmp_r5/sys.h, 99
sys_irq_enable
system/freertos/zynq7/sys.h, 97
system/freertos/zynqmp_r5/sys.h, 97
system/generic/zynq7/sys.h, 98
system/generic/zynqmp_r5/sys.h, 99
sys_irq_restore_enable
freertos/zynq7/sys.c, 109
freertos/zynqmp_r5/sys.c, 110
generic/zynq7/sys.c, 112
generic/zynqmp_r5/sys.c, 113
sys_irq_save_disable
freertos/zynq7/sys.c, 109
freertos/zynqmp_r5/sys.c, 110
generic/zynq7/sys.c, 112
generic/zynqmp_r5/sys.c, 113
sysfs_path
metal_state, 66
system/freertos/alloc.h
metal_allocate_memory, 67
metal_free_memory, 67
system/freertos/condition.h
metal_condition_broadcast, 73
metal_condition_init, 73
metal_condition_signal, 73
system/freertos/init.c
metal_irq_deinit, 82
metal_irq_init, 82
metal_sys_finish, 82
metal_sys_init, 82
system/freertos/io.c
metal_machine_io_mem_map, 84
system/freertos/irq.h
metal_irq_isr, 87
system/freertos/mutex.h
METAL_MUTEX_INIT, 91
metal_mutex_acquire, 91
metal_mutex_deinit, 91
metal_mutex_init, 91
metal_mutex_is_acquired, 91
metal_mutex_release, 91
metal_mutex_try_acquire, 91
system/freertos/zynq7/sys.h
MAX_IRQS, 96
sys_irq_disable, 97
sys_irq_enable, 97
system/freertos/zynqmp_r5/sys.h
MAX_IRQS, 97
sys_irq_disable, 97
sys_irq_enable, 97
system/generic/alloc.h
metal_allocate_memory, 68
metal_free_memory, 68
system/generic/condition.h
metal_condition_broadcast, 74
metal_condition_init, 74
metal_condition_signal, 74
system/generic/init.c
metal_irq_deinit, 83
metal_irq_init, 83
metal_sys_finish, 83
metal_sys_init, 83
system/generic/io.c
metal_machine_io_mem_map, 85
system/generic/irq.h
metal_irq_isr, 88
system/generic/mutex.h
METAL_MUTEX_INIT, 91
metal_mutex_acquire, 91
metal_mutex_deinit, 91
metal_mutex_init, 91
metal_mutex_is_acquired, 91
metal_mutex_release, 92
metal_mutex_try_acquire, 92
system/generic/zynq7/sys.h
MAX_IRQS, 98
sys_irq_disable, 98
sys_irq_enable, 98
system/generic/zynqmp_r5/sys.h
MAX_IRQS, 98
sys_irq_disable, 99
sys_irq_enable, 99
system/linux/alloc.h
metal_allocate_memory, 68
metal_free_memory, 68
system/linux/condition.h
metal_condition_broadcast, 74
metal_condition_init, 74
metal_condition_signal, 75
system/linux/device.c
for_each_linux_bus, 79
for_each_linux_driver, 79
linux_bus, 80

MAX_DRIVERS, 79
metal_linux_bus_close, 79
metal_linux_bus_finish, 79
metal_linux_bus_init, 79
metal_linux_bus_ops, 80
metal_linux_dev_close, 79
metal_linux_dev_dma_map, 79
metal_linux_dev_dma_unmap, 79
metal_linux_dev_irq_ack, 79
metal_linux_dev_open, 79
metal_linux_probe_bus, 79
metal_linux_probe_driver, 79
metal_linux_register_bus, 79
metal_uio_dev_bind, 79
metal_uio_dev_close, 79
metal_uio_dev_dma_map, 79
metal_uio_dev_dma_unmap, 79
metal_uio_dev_irq_ack, 79
metal_uio_dev_open, 79
metal_uio_read_map_attr, 79
to_linux_bus, 79
to_linux_device, 79
system/linux/init.c
metal_add_page_size, 83
metal_init_page_sizes, 83
metal_linux_irq_init, 83
metal_linux_irq_shutdown, 83
metal_pagesize_compare, 84
metal_sys_finish, 84
metal_sys_init, 84
system/linux/mutex.h
__metal_mutex_cmpxchg, 92
METAL_MUTEX_INIT, 92
metal_mutex_acquire, 92
metal_mutex_deinit, 92
metal_mutex_init, 92
metal_mutex_is_acquired, 92
metal_mutex_release, 92
metal_mutex_try_acquire, 92
system/linux/shmem.c
metal_shmem_io_close, 94
metal_shmem_io_ops, 94
metal_shmem_try_map, 94
system/linux/sys.h
MAX_PAGE_SIZES, 99
METAL_INVALID_VADDR, 99

TIME Interfaces, 47
metal_get_timestamp, 47
tmp_path
metal_state, 66
to_linux_bus
system/linux/device.c, 79
to_linux_device
system/linux/device.c, 79
Top Level Interfaces, 45
_metal, 46
METAL_BAD_IRQ, 45
METAL_BAD_OFFSET, 45
METAL_BAD_PHYS, 45
METAL_INIT_DEFAULTS, 45
metal_finish, 46
metal_init, 46
metal_irq_t, 46
metal_phys_addr_t, 46

utilities.c
metal_map, 114
metal_mktemp, 114
metal_mktemp_template, 114
metal_mktemp_unlinked, 114
metal_mlock, 115
metal_open, 115
metal_open_unlinked, 115
metal_randomize_string, 115
metal_unmap, 116
metal_virt2phys, 116

v
metal_condition, 57
metal_mutex_t, 62
metal_spinlock, 65
virt
IO Interfaces, 31
metal_sg, 64

waiters
metal_condition, 57
wakeups
metal_condition, 57
write
IO Interfaces, 31