Linux Xtensa Documentation

The kernel development community

Jan 15, 2023

CONTENTS

CHAPTER

ONE

ATOMIC OPERATION CONTROL (ATOMCTL) REGISTER

We Have Atomic Operation Control (ATOMCTL) Register. This register determines the effect of using a S32C1I instruction with various combinations of:

1. With and without an Coherent Cache Controller which can do Atomic Transactions to the memory internally.
2. With and without An Intelligent Memory Controller which can do Atomic Transactions itself.

The Core comes up with a default value of for the three types of cache ops:

```
0x28: (WB: Internal, WT: Internal, BY:Exception)
```

On the FPGA Cards we typically simulate an Intelligent Memory controller which can implement RCW transactions. For FPGA cards with an External Memory controller we let it to the atomic operations internally while doing a Cached (WB) transaction and use the Memory RCW for un-cached operations.

For systems without an coherent cache controller, non-MX, we always use the memory controllers RCW, thought non-MX controlers likely support the Internal Operation.

CUSTOMER-WARNING: Virtually all customers buy their memory controllers from vendors that don't support atomic RCW memory transactions and will likely want to configure this register to not use RCW.

Developers might find using RCW in Bypass mode convenient when testing with the cache being bypassed; for example studying cache alias problems.

See Section 4.3.12.4 of ISA; Bits:

```
  WB      WT       BY
5   4 | 3   2 | 1   0
```

2 Bit			
Field			
Values	WB - Write Back	WT - Write Thru	BY - Bypass
0	Exception	Exception	Exception
1	RCW Transaction	RCW Transaction	RCW Transaction
2	Internal Operation	Internal Operation	Reserved
3	Reserved	Reserved	Reserved

CHAPTER

TWO

PASSING BOOT PARAMETERS TO THE KERNEL

Boot parameters are represented as a TLV list in the memory. Please see arch/xtensa/include/asm/bootparam.h for definition of the bp_tag structure and tag value constants. First entry in the list must have type BP_TAG_FIRST, last entry must have type BP_TAG_LAST. The address of the first list entry is passed to the kernel in the register a2. The address type depends on MMU type:

- For configurations without MMU, with region protection or with MPU the address must be the physical address.
- For configurations with region translarion MMU or with MMUv3 and CONFIG_MMU=n the address must be a valid address in the current mapping. The kernel will not change the mapping on its own.
- For configurations with MMUv2 the address must be a virtual address in the default virtual mapping (0xd0000000..0xffffffff).
- For configurations with MMUv3 and CONFIG_MMU=y the address may be either a virtual or physical address. In either case it must be within the default virtual mapping. It is considered physical if it is within the range of physical addresses covered by the default KSEG mapping (XCHAL_KSEG_PADDR.. XCHAL_KSEG_PADDR + XCHAL_KSEG_SIZE), otherwise it is considered virtual.

CHAPTER

THREE

MMUV3 INITIALIZATION SEQUENCE

The code in the initialize_mmu macro sets up MMUv3 memory mapping identically to MMUv2 fixed memory mapping. Depending on CONFIG_INITIALIZE_XTENSA_MMU_INSIDE_VMLINUX symbol this code is located in addresses it was linked for (symbol undefined), or not (symbol defined), so it needs to be position-independent.

The code has the following assumptions:

- This code fragment is run only on an MMU v3.
- TLBs are in their reset state.
- ITLBCFG and DTLBCFG are zero (reset state).
- RASID is 0x04030201 (reset state).
- PS.RING is zero (reset state).
- LITBASE is zero (reset state, PC-relative literals); required to be PIC.

TLB setup proceeds along the following steps.

Legend:

- VA = virtual address (two upper nibbles of it);
- PA = physical address (two upper nibbles of it);
- pc = physical range that contains this code;

After step 2, we jump to virtual address in the range 0x40000000..0x5fffffff or 0x00000000..0x1fffffff, depending on whether the kernel was loaded below 0x40000000 or above. That address corresponds to next instruction to execute in this code. After step 4, we jump to intended (linked) address of this code. The scheme below assumes that the kernel is loaded below 0x40000000.

•	Step0	Step1	Step2	Step3		Step4	Step5
VA	PA	PA	PA	PA	VA	PA	PA
E0..FF	-> E0	-> E0	-> E0		F0..FF	-> F0	-> F0
C0..DF	-> C0	-> C0	-> C0		E0..EF	-> F0	-> F0
A0..BF	-> A0	-> A0	-> A0		D8..DF	-> 00	-> 00
80..9F	-> 80	-> 80	-> 80		D0..D7	-> 00	-> 00
60..7F	-> 60	-> 60	-> 60				
40..5F	-> 40		-> pc	-> pc	40..5F	-> pc	
20..3F	-> 20	-> 20	-> 20				
00..1F	-> 00	-> 00	-> 00				

The default location of IO peripherals is above 0xf0000000. This may be changed using a "ranges" property in a device tree simple-bus node. See the Devicetree Specification, section 4.5 for details on the syntax and semantics of simple-bus nodes. The following limitations apply:

1. Only top level simple-bus nodes are considered
2. Only one (first) simple-bus node is considered
3. Empty "ranges" properties are not supported
4. Only the first triplet in the "ranges" property is considered
5. The parent-bus-address value is rounded down to the nearest 256MB boundary
6. The IO area covers the entire 256MB segment of parent-bus-address; the "ranges" triplet length field is ignored

3.1 MMUv3 address space layouts.

Default MMUv2-compatible layout:

```
                    Symbol                      VADDR       Size
+------------------+
| Userspace        |                            0x00000000  TASK_SIZE
+------------------+                            0x40000000
+------------------+
| Page table       |  XCHAL_PAGE_TABLE_VADDR    0x80000000  XCHAL_PAGE_TABLE_
 ↪SIZE
+------------------+
| KASAN shadow map |  KASAN_SHADOW_START        0x80400000  KASAN_SHADOW_SIZE
+------------------+                            0x8e400000
+------------------+
| VMALLOC area     |  VMALLOC_START             0xc0000000  128MB - 64KB
+------------------+  VMALLOC_END
+------------------+
| Cache aliasing   |  TLBTEMP_BASE_1            0xc8000000  DCACHE_WAY_SIZE
| remap area 1     |
+------------------+
```

```
| Cache aliasing   |  TLBTEMP_BASE_2                        DCACHE_WAY_SIZE
| remap area 2     |
+------------------+
+------------------+
| KMAP area        |  PKMAP_BASE                            PTRS_PER_PTE *
|                  |                                        DCACHE_N_COLORS *
|                  |                                        PAGE_SIZE
|                  |                                        (4MB * DCACHE_N_
 ↪COLORS)
+------------------+
| Atomic KMAP area |  FIXADDR_START                         KM_TYPE_NR *
|                  |                                        NR_CPUS *
|                  |                                        DCACHE_N_COLORS *
|                  |                                        PAGE_SIZE
+------------------+  FIXADDR_TOP              0xcffff000
+------------------+
| Cached KSEG      |  XCHAL_KSEG_CACHED_VADDR  0xd0000000  128MB
+------------------+
| Uncached KSEG    |  XCHAL_KSEG_BYPASS_VADDR  0xd8000000  128MB
+------------------+
| Cached KIO       |  XCHAL_KIO_CACHED_VADDR   0xe0000000  256MB
+------------------+
| Uncached KIO     |  XCHAL_KIO_BYPASS_VADDR   0xf0000000  256MB
+------------------+
```

256MB cached + 256MB uncached layout:

```
                     Symbol                    VADDR       Size
+------------------+
| Userspace        |                           0x00000000  TASK_SIZE
+------------------+                           0x40000000
+------------------+
| Page table       |  XCHAL_PAGE_TABLE_VADDR   0x80000000  XCHAL_PAGE_TABLE_
 ↪SIZE
+------------------+
| KASAN shadow map |  KASAN_SHADOW_START       0x80400000  KASAN_SHADOW_SIZE
+------------------+                           0x8e400000
+------------------+
| VMALLOC area     |  VMALLOC_START            0xa0000000  128MB - 64KB
+------------------+  VMALLOC_END
+------------------+
| Cache aliasing   |  TLBTEMP_BASE_1           0xa8000000  DCACHE_WAY_SIZE
| remap area 1     |
+------------------+
| Cache aliasing   |  TLBTEMP_BASE_2                       DCACHE_WAY_SIZE
| remap area 2     |
+------------------+
+------------------+
| KMAP area        |  PKMAP_BASE                           PTRS_PER_PTE *
|                  |                                       DCACHE_N_COLORS *
|                  |                                       PAGE_SIZE
```

```
|                  |                                          (4MB * DCACHE_N_
↪COLORS)
+------------------+
| Atomic KMAP area |  FIXADDR_START                           KM_TYPE_NR *
|                  |                                          NR_CPUS *
|                  |                                          DCACHE_N_COLORS *
|                  |                                          PAGE_SIZE
+------------------+  FIXADDR_TOP               0xafffe000
+------------------+
| Cached KSEG      |  XCHAL_KSEG_CACHED_VADDR   0xb0000000  256MB
+------------------+
| Uncached KSEG    |  XCHAL_KSEG_BYPASS_VADDR   0xc0000000  256MB
+------------------+
+------------------+
| Cached KIO       |  XCHAL_KIO_CACHED_VADDR    0xe0000000  256MB
+------------------+
| Uncached KIO     |  XCHAL_KIO_BYPASS_VADDR    0xf0000000  256MB
+------------------+
```

512MB cached + 512MB uncached layout:

```
                     Symbol                    VADDR       Size
+------------------+
| Userspace        |                           0x00000000  TASK_SIZE
+------------------+                           0x40000000
+------------------+
| Page table       |  XCHAL_PAGE_TABLE_VADDR   0x80000000  XCHAL_PAGE_TABLE_
↪SIZE
+------------------+
| KASAN shadow map |  KASAN_SHADOW_START       0x80400000  KASAN_SHADOW_SIZE
+------------------+                           0x8e400000
+------------------+
| VMALLOC area     |  VMALLOC_START            0x90000000  128MB - 64KB
+------------------+  VMALLOC_END
+------------------+
| Cache aliasing   |  TLBTEMP_BASE_1           0x98000000  DCACHE_WAY_SIZE
| remap area 1     |
+------------------+
| Cache aliasing   |  TLBTEMP_BASE_2                       DCACHE_WAY_SIZE
| remap area 2     |
+------------------+
+------------------+
| KMAP area        |  PKMAP_BASE                           PTRS_PER_PTE *
|                  |                                       DCACHE_N_COLORS *
|                  |                                       PAGE_SIZE
|                  |                                       (4MB * DCACHE_N_
↪COLORS)
+------------------+
| Atomic KMAP area |  FIXADDR_START                        KM_TYPE_NR *
|                  |                                       NR_CPUS *
|                  |                                       DCACHE_N_COLORS *
```

			PAGE_SIZE
	FIXADDR_TOP	0x9ffff000	
Cached KSEG	XCHAL_KSEG_CACHED_VADDR	0xa0000000	512MB
Uncached KSEG	XCHAL_KSEG_BYPASS_VADDR	0xc0000000	512MB
Cached KIO	XCHAL_KIO_CACHED_VADDR	0xe0000000	256MB
Uncached KIO	XCHAL_KIO_BYPASS_VADDR	0xf0000000	256MB

CHAPTER

FOUR

FEATURE STATUS ON XTENSA ARCHITECTURE

Subsystem	Feature	Kconfig	Status	Desc
core	cBPF-JIT	HAVE_CBPF_JIT	TODO	arch
core	eBPF-JIT	HAVE_EBPF_JIT	TODO	arch
core	generic-idle-thread	GENERIC_SMP_IDLE_THREAD	ok	arch
core	jump-labels	HAVE_ARCH_JUMP_LABEL	ok	arch
core	thread-info-in-task	THREAD_INFO_IN_TASK	TODO	arch
core	tracehook	HAVE_ARCH_TRACEHOOK	ok	arch
debug	debug-vm-pgtable	ARCH_HAS_DEBUG_VM_PGTABLE	ok	arch
debug	gcov-profile-all	ARCH_HAS_GCOV_PROFILE_ALL	TODO	arch
debug	KASAN	HAVE_ARCH_KASAN	ok	arch
debug	kcov	ARCH_HAS_KCOV	TODO	arch
debug	kgdb	HAVE_ARCH_KGDB	TODO	arch
debug	kmemleak	HAVE_DEBUG_KMEMLEAK	ok	arch
debug	kprobes	HAVE_KPROBES	TODO	arch
debug	kprobes-on-ftrace	HAVE_KPROBES_ON_FTRACE	TODO	arch
debug	kretprobes	HAVE_KRETPROBES	TODO	arch
debug	optprobes	HAVE_OPTPROBES	TODO	arch
debug	stackprotector	HAVE_STACKPROTECTOR	ok	arch
debug	uprobes	ARCH_SUPPORTS_UPROBES	TODO	arch
debug	user-ret-profiler	HAVE_USER_RETURN_NOTIFIER	TODO	arch
io	dma-contiguous	HAVE_DMA_CONTIGUOUS	ok	arch
locking	cmpxchg-local	HAVE_CMPXCHG_LOCAL	TODO	arch
locking	lockdep	LOCKDEP_SUPPORT	ok	arch
locking	queued-rwlocks	ARCH_USE_QUEUED_RWLOCKS	ok	arch
locking	queued-spinlocks	ARCH_USE_QUEUED_SPINLOCKS	ok	arch
perf	kprobes-event	HAVE_REGS_AND_STACK_ACCESS_API	TODO	arch
perf	perf-regs	HAVE_PERF_REGS	TODO	arch
perf	perf-stackdump	HAVE_PERF_USER_STACK_DUMP	TODO	arch
sched	membarrier-sync-core	ARCH_HAS_MEMBARRIER_SYNC_CORE	TODO	arch
sched	numa-balancing	ARCH_SUPPORTS_NUMA_BALANCING	—	arch
seccomp	seccomp-filter	HAVE_ARCH_SECCOMP_FILTER	ok	arch
time	arch-tick-broadcast	ARCH_HAS_TICK_BROADCAST	TODO	arch
time	clockevents	!LEGACY_TIMER_TICK	ok	arch
time	context-tracking	HAVE_CONTEXT_TRACKING	ok	arch
time	irq-time-acct	HAVE_IRQ_TIME_ACCOUNTING	ok	arch
time	virt-cpuacct	HAVE_VIRT_CPU_ACCOUNTING	ok	arch

Table 1 – continued from p

Subsystem	Feature	Kconfig	Status	Desc
vm	batch-unmap-tlb-flush	ARCH_WANT_BATCHED_UNMAP_TLB_FLUSH	TODO	arch
vm	ELF-ASLR	ARCH_HAS_ELF_RANDOMIZE	TODO	arch
vm	huge-vmap	HAVE_ARCH_HUGE_VMAP	TODO	arch
vm	ioremap_prot	HAVE_IOREMAP_PROT	TODO	arch
vm	PG_uncached	ARCH_USES_PG_UNCACHED	TODO	arch
vm	pte_special	ARCH_HAS_PTE_SPECIAL	TODO	arch
vm	THP	HAVE_ARCH_TRANSPARENT_HUGEPAGE	—	arch