ICH_LR<n>_EL2, Interrupt Controller List Registers, n = 0 - 15

The ICH_LR<n>_EL2 characteristics are:

Purpose

Provides interrupt context information for the virtual CPU interface.

Configuration

AArch64 System register ICH_LR<n>_EL2 bits [31:0] are architecturally mapped to AArch32 System register ICH_LR<n>[31:0].

AArch64 System register ICH_LR<n>_EL2 bits [63:32] are architecturally mapped to AArch32 System register ICH_LRC<n>[31:0].

This register is present only when FEAT_GICv3 is implemented and (EL2 is implemented or EL3 is implemented). Otherwise, direct accesses to ICH_LR<n>_EL2 are undefined.

If EL2 is not implemented, this register is res0 from EL3.

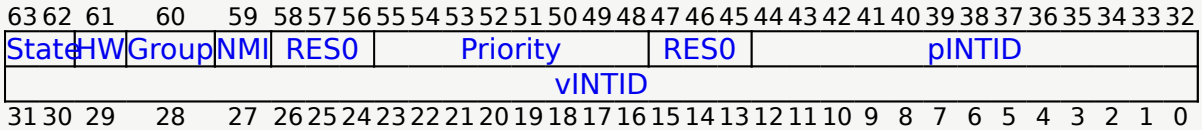
If list register n is not implemented, then accesses to this register are undefined.

This register has no effect if EL2 is not enabled in the current Security state.

Attributes

ICH_LR<n>_EL2 is a 64-bit register.

Field descriptions



State, bits [63:62]

The state of the interrupt:

State	Meaning
0b00	Invalid (Inactive).

0b01	Pending.
0b10	Active.
0b11	Pending and active.

The GIC updates these state bits as virtual interrupts proceed through the interrupt life cycle. Entries in the invalid state are ignored, except for the purpose of generating virtual maintenance interrupts.

For hardware interrupts, the pending and active state is held in the physical Distributor rather than the virtual CPU interface. A hypervisor must only use the pending and active state for software originated interrupts, which are typically associated with virtual devices, or SGIs.

The reset behavior of this field is:

- On a Warm reset, this field resets to an architecturally unknown value.

HW, bit [61]

Indicates whether this virtual interrupt maps directly to a hardware interrupt, meaning that it corresponds to a physical interrupt. Deactivation of the virtual interrupt also causes the deactivation of the physical interrupt with the ID that the pINTID field indicates.

HW	Meaning
0b0	The interrupt is triggered entirely by software. No notification is sent to the Distributor when the virtual interrupt is deactivated.
0b1	The interrupt maps directly to a hardware interrupt. A deactivate interrupt request is sent to the Distributor when the virtual interrupt is deactivated, using the pINTID field from this register to indicate the physical interrupt ID. If ICH_VMCR_EL2.VEOIM is 0, this request corresponds to a write to ICC_EOIR0_EL1 or ICC_EOIR1_EL1. Otherwise, it corresponds to a write to ICC_DIR_EL1.

The reset behavior of this field is:

- On a Warm reset, this field resets to an architecturally unknown value.

Group, bit [60]

Indicates the group for this virtual interrupt.

Group	Meaning
0b0	This is a Group 0 virtual interrupt. ICH_VMCR_EL2.VFIQEn determines whether it is signaled as a virtual IRQ or as a virtual FIQ, and ICH_VMCR_EL2.VENG0 enables signaling of this interrupt to the virtual machine.
0b1	This is a Group 1 virtual interrupt, signaled as a virtual IRQ. ICH_VMCR_EL2.VENG1 enables the signalling of this interrupt to the virtual machine. If ICH_VMCR_EL2.VCBPR is 0, then ICC_BPR1_EL1 determines if a pending Group 1 interrupt has sufficient priority to preempt current execution. Otherwise, ICH_LR<n>_EL2 determines preemption.

The reset behavior of this field is:

- On a Warm reset, this field resets to an architecturally unknown value.

NMI, bit [59]
When FEAT_GICv3_NMI is implemented:

Indicates whether the virtual priority has the non-maskable property.

NMI	Meaning
0b0	vINTID does not have the non-maskable interrupt property.
0b1	vINTID has the non-maskable interrupt property.

Setting ICH_LR<n>_EL2.NMI to 1 when ICH_LR<n>_EL2.State is not Invalid is CONSTRAINTED unpredictable if either ICH_LR<n>_EL2.vINTID indicates an LPI or ICH_LR<n>_EL2.Group is 0.

The permitted behaviors are:

- ICH_LR<n>_EL2.NMI is treated as 0 for all purposes other than a direct read of the register.
- The virtual interrupt is presented with superpriority.

The reset behavior of this field is:

- On a Warm reset, this field resets to an architecturally unknown value.

Otherwise:

Reserved, res0.

Bits [58:56]

Reserved, res0.

Priority, bits [55:48]

The priority of this interrupt.

It is implementation defined how many bits of priority are implemented, though at least five bits must be implemented. Unimplemented bits are res0 and start from bit[48] up to bit[50]. The number of implemented bits can be discovered from ICH_VTR_EL2.PRIbits.

When ICH_LR<n>_EL2.NMI is set to 1, this field is res0 and the virtual interrupt's priority is treated as `0x00`.

The reset behavior of this field is:

- On a Warm reset, this field resets to an architecturally unknown value.

Bits [47:45]

Reserved, res0.

pINTID, bits [44:32]

Physical INTID, for hardware interrupts.

When ICH_LR<n>_EL2.HW is 0 (there is no corresponding physical interrupt), this field has the following meaning:

- Bits[44:42] : res0.
- Bit[41] : EOI. If this bit is 1, then when the interrupt identified by vINTID is deactivated, a maintenance interrupt is asserted.
- Bits[40:32] : res0.

When ICH_LR<n>_EL2.HW is 1 (there is a corresponding physical interrupt):

- This field indicates the physical INTID. This field is only required to implement enough bits to hold a valid value for the implemented INTID size. Any unused higher order bits are res0.
- When ICC_CTLR_EL1.ExtRange is 0, then bits[44:42] of this field are res0.
- If the value of pINTID is not a vald INTID, behavior is unpredictable. If the value of pINTID indicates a PPI, this field applies to the PPI associated with this same physical PE ID as the virtual CPU interface requesting the deactivation.

A hardware physical identifier is only required in List Registers for interrupts that require deactivation. This means only 13 bits of Physical INTID are required, regardless of the number specified by ICC_CTLR_EL1.IDbits.

The reset behavior of this field is:

- On a Warm reset, this field resets to an architecturally unknown value.

vINTID, bits [31:0]

Virtual INTID of the interrupt.

If the value of vINTID is 1020-1023 and ICH_LR<n>_EL2.State! =`0b00` (Inactive), behavior is unpredictable.

Behavior is unpredictable if two or more List Registers specify the same vINTID when:

- ICH_LR<n>_EL2.State == `0b01`.
- ICH_LR<n>_EL2.State == `0b10`.
- ICH_LR<n>_EL2.State == `0b11`.

It is implementation defined how many bits are implemented, though at least 16 bits must be implemented. Unimplemented bits are res0. The number of implemented bits can be discovered from ICH_VTR_EL2.IDbits.

When ICC_SRE_EL1.SRE == 0, specifying a vINTID in the LPI range is unpredictable

---

Note

When a VM is using memory-mapped access to the GIC, software must ensure that the correct source PE ID is provided in bits[12:10].

The reset behavior of this field is:

- On a Warm reset, this field resets to an architecturally unknown value.

Accessing ICH_LR<n>_EL2

Accesses to this register use the following encodings in the System register encoding space:

MRS <Xt>, ICH_LR<m>_EL2 ; Where m = 0-15

op0	op1	CRn	CRm	op2
0b11	0b100	0b1100	0b110:m[3]	m[2:0]

```
integer m = UInt(CRm<0>:op2<2:0>);

if m >= NUM_GIC_LIST_REGS then
    UNDEFINED;
elsif PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.<NV2,NV> == '11' then
        X[t, 64] = NVMem[0x400 + (8 * m)];
    elsif EL2Enabled() && HCR_EL2.NV == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    if ICC_SRE_EL2.SRE == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = ICH_LR_EL2[m];
elsif PSTATE.EL == EL3 then
    if ICC_SRE_EL3.SRE == '0' then
        AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = ICH_LR_EL2[m];
```

MSR ICH_LR<m>_EL2, <Xt> ; Where m = 0-15

op0	op1	CRn	CRm	op2
0b11	0b100	0b1100	0b110:m[3]	m[2:0]

```
integer m = UInt(CRm<0>:op2<2:0>);
```

```
if m >= NUM_GIC_LIST_REGS then
    UNDEFINED;
elsif PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.<NV2,NV> == '11' then
        NVMem[0x400 + (8 * m)] = X[t, 64];
    elsif EL2Enabled() && HCR_EL2.NV == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    if ICC_SRE_EL2.SRE == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        ICH_LR_EL2[m] = X[t, 64];
elsif PSTATE.EL == EL3 then
    if ICC_SRE_EL3.SRE == '0' then
        AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ICH_LR_EL2[m] = X[t, 64];
```

28/03/2023 16:01; 72747e43966d6b97dcbd230a1b3f0421d1ea3d94