ARM Virtual Generic Interrupt Controller v3 and later (VGICv3)

Device types supported:

- KVM_DEV_TYPE_ARM_VGIC_V3 ARM Generic Interrupt Controller v3.0

Only one VGIC instance may be instantiated through this API. The created VGIC will act as the VM interrupt controller, requiring emulated user-space devices to inject interrupts to the VGIC instead of directly to CPUs. It is not possible to create both a GICv3 and GICv2 on the same VM.

Creating a guest GICv3 device requires a host GICv3 as well.

Groups:

KVM_DEV_ARM_VGIC_GRP_ADDR

Attributes:

KVM_VGIC_V3_ADDR_TYPE_DIST (rw, 64-bit)

Base address in the guest physical address space of the GICv3 distributor register mappings. Only valid for KVM_DEV_TYPE_ARM_VGIC_V3. This address needs to be 64K aligned and the region covers 64 KByte.

KVM_VGIC_V3_ADDR_TYPE_REDIST (rw, 64-bit)

Base address in the guest physical address space of the GICv3 redistributor register mappings. There are two 64K pages for each VCPU and all of the redistributor pages are contiguous. Only valid for KVM_DEV_TYPE_ARM_VGIC_V3. This address needs to be 64K aligned.

KVM_VGIC_V3_ADDR_TYPE_REDIST_REGION (rw, 64-bit)

The attribute data pointed to by kvm_device_attr.addr is a __u64 value:

```
bits:     | 63   ....  52  |  51   ....   16 | 15 - 12  |11 - 0
values:   |     count      |       base      |  flags   | index
```

- index encodes the unique redistributor region index
- flags: reserved for future use, currently 0
- base field encodes bits [51:16] of the guest physical base address of the first redistributor in the region.
- count encodes the number of redistributors in the region. Must be greater than 0.

There are two 64K pages for each redistributor in the region and redistributors are laid out contiguously within the region. Regions are filled with redistributors in the index order. The sum of all region count fields must be greater than or equal to the number of VCPUs. Redistributor regions must be registered in the incremental index order, starting from index 0.

The characteristics of a specific redistributor region can be read by presetting the index field in the attr data. Only valid for KVM_DEV_TYPE_ARM_VGIC_V3.

It is invalid to mix calls with KVM_VGIC_V3_ADDR_TYPE_REDIST and KVM_VGIC_V3_ADDR_TYPE_REDIST_REGION attributes.

Errors:

-E2BIG	Address outside of addressable IPA range
-EINVAL	Incorrectly aligned address, bad redistributor region count/index, mixed redistributor region attribute usage
-EEXIST	Address already configured
-ENOENT	Attempt to read the characteristics of a non existing redistributor region
-ENXIO	The group or attribute is unknown/unsupported for this device or hardware support is missing.
-EFAULT	Invalid user pointer for attr->addr.

KVM_DEV_ARM_VGIC_GRP_DIST_REGS, KVM_DEV_ARM_VGIC_GRP_REDIST_REGS

Attributes:

The attr field of kvm_device_attr encodes two values:

```
bits:     | 63   ....  32  |  31   ....    0 |
values:   |      mpidr     |      offset     |
```

All distributor regs are (rw, 32-bit) and kvm_device_attr.addr points to a __u32 value. 64-bit registers must be accessed by separately accessing the lower and higher word.

Writes to read-only registers are ignored by the kernel.

KVM_DEV_ARM_VGIC_GRP_DIST_REGS accesses the main distributor registers. KVM_DEV_ARM_VGIC_GRP_REDIST_REGS accesses the redistributor of the CPU specified by the mpidr.

The offset is relative to the "[Re]Distributor base address" as defined in the GICv3/4 specs. Getting or setting such a register has the same effect as reading or writing the register on real hardware, except for the following registers: GICD_STATUSR, GICR_STATUSR, GICD_ISPENDR, GICR_ISPENDR0, GICD_ICPENDR, and GICR_ICPENDR0. These registers behave differently when accessed via this interface compared to their architecturally defined behavior to allow software a full view of the VGIC's internal state.

The mpidr field is used to specify which redistributor is accessed. The mpidr is ignored for the distributor.

The mpidr encoding is based on the affinity information in the architecture defined MPIDR, and the field is encoded as follows:

```
| 63 .... 56 | 55 .... 48 | 47 .... 40 | 39 .... 32 |
|    Aff3    |    Aff2    |    Aff1    |    Aff0    |
```

Note that distributor fields are not banked, but return the same value regardless of the mpidr used to access the register.

GICD_IIDR.Revision is updated when the KVM implementation is changed in a way directly observable by the guest or userspace. Userspace should read GICD_IIDR from KVM and write back the read value to confirm its expected behavior is aligned with the KVM implementation. Userspace should set GICD_IIDR before setting any other registers to ensure the expected behavior.

The GICD_STATUSR and GICR_STATUSR registers are architecturally defined such that a write of a clear bit has no effect, whereas a write with a set bit clears that value. To allow userspace to freely set the values of these two registers, setting the attributes with the register offsets for these two registers simply sets the non-reserved bits to the value written.

Accesses (reads and writes) to the GICD_ISPENDR register region and GICR_ISPENDR0 registers get/set the value of the latched pending state for the interrupts.

This is identical to the value returned by a guest read from ISPENDR for an edge triggered interrupt, but may differ for level triggered interrupts. For edge triggered interrupts, once an interrupt becomes pending (whether because of an edge detected on the input line or because of a guest write to ISPENDR) this state is "latched", and only cleared when either the interrupt is activated or when the guest writes to ICPENDR. A level triggered interrupt may be pending either because the level input is held high by a device, or because of a guest write to the ISPENDR register. Only ISPENDR writes are latched; if the device lowers the line level then the interrupt is no longer pending unless the guest also wrote to ISPENDR, and conversely writes to ICPENDR or activations of the interrupt do not clear the pending status if the line level is still being held high. (These rules are documented in the GICv3 specification descriptions of the ICPENDR and ISPENDR registers.) For a level triggered interrupt the value accessed here is that of the latch which is set by ISPENDR and cleared by ICPENDR or interrupt activation, whereas the value returned by a guest read from ISPENDR is the logical OR of the latch value and the input line level.

Raw access to the latch state is provided to userspace so that it can save and restore the entire GIC internal state (which is defined by the combination of the current input line level and the latch state, and cannot be deduced from purely the line level and the value of the ISPENDR registers).

Accesses to GICD_ICPENDR register region and GICR_ICPENDR0 registers have RAZ/WI semantics, meaning that reads always return 0 and writes are always ignored.

Errors:

-ENXIO	Getting or setting this register is not yet supported
-EBUSY	One or more VCPUs are running

KVM_DEV_ARM_VGIC_GRP_CPU_SYSREGS

Attributes:

The attr field of kvm_device_attr encodes two values:

```
bits:     | 63      ....       32 | 31  ....  16 | 15  ....  0 |
values:   |         mpidr         |      RES     |    instr    |
```

The mpidr field encodes the CPU ID based on the affinity information in the architecture defined MPIDR, and the field is encoded as follows:

```
| 63 .... 56 | 55 .... 48 | 47 .... 40 | 39 .... 32 |
|    Aff3    |    Aff2    |    Aff1    |    Aff0    |
```

The instr field encodes the system register to access based on the fields defined in the A64 instruction set encoding for system register access (RES means the bits are reserved for future use and should be zero):

```
| 15 ... 14 | 13 ... 11 | 10 ... 7 | 6 ... 3 | 2 ... 0 |
|   Op 0    |    Op1    |    CRn   |   CRm   |   Op2   |
```

All system regs accessed through this API are (rw, 64-bit) and kvm_device_attr.addr points to a __u64 value.

KVM_DEV_ARM_VGIC_GRP_CPU_SYSREGS accesses the CPU interface registers for the CPU specified by the mpidr field.

CPU interface registers access is not implemented for AArch32 mode. Error -ENXIO is returned when accessed in AArch32 mode.

Errors:

-ENXIO	Getting or setting this register is not yet supported
-EBUSY	VCPU is running
-EINVAL	Invalid mpidr or register value supplied

KVM_DEV_ARM_VGIC_GRP_NR_IRQS

Attributes:

A value describing the number of interrupts (SGI, PPI and SPI) for this GIC instance, ranging from 64 to 1024, in increments of 32.

kvm_device_attr.addr points to a __u32 value.

Errors:

-EINVAL	Value set is out of the expected range
-EBUSY	Value has already be set.

KVM_DEV_ARM_VGIC_GRP_CTRL

Attributes:

KVM_DEV_ARM_VGIC_CTRL_INIT

request the initialization of the VGIC, no additional parameter in kvm_device_attr.addr. Must be called after all VCPUs have been created.

KVM_DEV_ARM_VGIC_SAVE_PENDING_TABLES

save all LPI pending bits into guest RAM pending tables.

The first kB of the pending table is not altered by this operation.

Errors:

-ENXIO	VGIC not properly configured as required prior to calling this attribute
-ENODEV	no online VCPU
-ENOMEM	memory shortage when allocating vgic internal data
-EFAULT	Invalid guest ram access
-EBUSY	One or more VCPUS are running

KVM_DEV_ARM_VGIC_GRP_LEVEL_INFO

Attributes:

The attr field of kvm_device_attr encodes the following values:

```
bits:     | 63      ....       32 | 31   ....    10 | 9  ....  0 |
values:   |         mpidr         |      info       |   vINTID   |
```

The vINTID specifies which set of IRQs is reported on.

The info field specifies which information userspace wants to get or set using this interface. Currently we support the following info values:

VGIC_LEVEL_INFO_LINE_LEVEL:

Get/Set the input level of the IRQ line for a set of 32 contiguously numbered interrupts.

vINTID must be a multiple of 32.

kvm_device_attr.addr points to a __u32 value which will contain a bitmap where a set bit means the interrupt level is asserted.

Bit[n] indicates the status for interrupt vINTID + n.

SGIs and any interrupt with a higher ID than the number of interrupts supported, will be RAZ/WI. LPIs are always edge-triggered and are therefore not supported by this interface.

PPIs are reported per VCPU as specified in the mpidr field, and SPIs are reported with the same value regardless of the mpidr specified.

The mpidr field encodes the CPU ID based on the affinity information in the architecture defined MPIDR, and the field is encoded as follows:

```
| 63 .... 56 | 55 .... 48 | 47 .... 40 | 39 .... 32 |
|    Aff3    |    Aff2    |    Aff1    |    Aff0    |
```

Errors:

-EINVAL	vINTID is not multiple of 32 or info field is not VGIC_LEVEL_INFO_LINE_LEVEL