[AArch32 Registers] [AArch64 Registers] [AArch32 Instructions] [AArch64 Instructions] [Index by Encoding] [External Registers]

SPSR_irq, Saved Program Status Register (IRQ mode)

The SPSR_irq characteristics are:

Purpose

Holds the saved process state when an exception is taken to IRQ mode.

Configuration

AArch64 System register SPSR_irq bits [31:0] are architecturally mapped to AArch32 System register SPSR_irq[31:0].

If EL1 only supports execution in AArch64 state, this register is res0 from EL2 and EL3.

Attributes

SPSR_irq is a 64-bit register.

Field descriptions

When EL1 can only use AArch64:

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
RES0																															
RES0																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Bits [63:0]

Reserved, res0.

Otherwise:

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
RES0																															
N	Z	C	V	Q	IT[1:0]		J	SSBS	PAN	DIT	IL	GE				IT[7:2]						E	A	I	F	T	M[4:0]				
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Bits [63:32]

Reserved, res0.

N, bit [31]

Negative Condition flag. Set to the value of PSTATE.N on taking an exception to IRQ mode, and copied to PSTATE.N on executing an illegal exception return operation in IRQ mode.

The reset behavior of this field is:

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

Z, bit [30]

Zero Condition flag. Set to the value of PSTATE.Z on taking an exception to IRQ mode, and copied to PSTATE.Z on executing an illegal exception return operation in IRQ mode.

The reset behavior of this field is:

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

C, bit [29]

Carry Condition flag. Set to the value of PSTATE.C on taking an exception to IRQ mode, and copied to PSTATE.C on executing an illegal exception return operation in IRQ mode.

The reset behavior of this field is:

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

V, bit [28]

Overflow Condition flag. Set to the value of PSTATE.V on taking an exception to IRQ mode, and copied to PSTATE.V on executing an illegal exception return operation in IRQ mode.

The reset behavior of this field is:

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

Q, bit [27]

Overflow or saturation flag. Set to the value of PSTATE.Q on taking an exception to IRQ mode, and copied to PSTATE.Q on executing an illegal exception return operation in IRQ mode.

The reset behavior of this field is:

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

IT, bits [15:10, 26:25]

If-Then. Set to the value of PSTATE.IT on taking an exception to IRQ mode, and copied to PSTATE.IT on executing an illegal exception return operation in IRQ mode.

SPSR_irq.IT must contain a value that is valid for the instruction being returned to.

The IT field is split as follows:

- IT[1:0] is SPSR_irq[26:25].
- IT[7:2] is SPSR_irq[15:10].

The reset behavior of this field is:

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

J, bit [24]

res0.

In previous versions of the architecture, the {J, T} bits determined the AArch32 Instruction set state.

Armv8 does not support either Jazelle state or T32EE state, and the T bit determines the Instruction set state.

SSBS, bit [23]

When FEAT_SSBS is implemented:

Speculative Store Bypass. Set to the value of PSTATE.SSBS on taking an exception to IRQ mode, and copied to PSTATE.SSBS on executing an illegal exception return operation in IRQ mode.

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

PAN, bit [22]
When FEAT_PAN is implemented:

Privileged Access Never. Set to the value of PSTATE.PAN on taking an exception to IRQ mode, and copied to PSTATE.PAN on executing an illegal exception return operation in IRQ mode.

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

DIT, bit [21]
When FEAT_DIT is implemented:

Data Independent Timing. Set to the value of PSTATE.DIT on taking an exception to IRQ mode, and copied to PSTATE.DIT on executing an illegal exception return operation in IRQ mode.

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

IL, bit [20]

Illegal Execution state. Set to the value of PSTATE.IL on taking an exception to IRQ mode, and copied to PSTATE.IL on executing an illegal exception return operation in IRQ mode.

The reset behavior of this field is:

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

GE, bits [19:16]

Greater than or Equal flags. Set to the value of PSTATE.GE on taking an exception to IRQ mode, and copied to PSTATE.GE on executing an illegal exception return operation in IRQ mode.

The reset behavior of this field is:

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

E, bit [9]

Endianness. Set to the value of PSTATE.E on taking an exception to IRQ mode, and copied to PSTATE.E on executing an illegal exception return operation in IRQ mode.

If the implementation does not support big-endian operation, SPSR_irq.E is res0. If the implementation does not support little-endian operation, SPSR_irq.E is res1. On executing an illegal exception return operation in IRQ mode, if the implementation does not support big-endian operation at the Exception level being returned to, SPSR_irq.E is res0, and if the implementation does not support little-endian operation at the Exception level being returned to, SPSR_irq.E is res1.

The reset behavior of this field is:

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

A, bit [8]

SError interrupt mask. Set to the value of PSTATE.A on taking an exception to IRQ mode, and copied to PSTATE.A on executing an illegal exception return operation in IRQ mode.

The reset behavior of this field is:

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

I, bit [7]

IRQ interrupt mask. Set to the value of PSTATE.I on taking an exception to IRQ mode, and copied to PSTATE.I on executing an illegal exception return operation in IRQ mode.

The reset behavior of this field is:

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

F, bit [6]

FIQ interrupt mask. Set to the value of PSTATE.F on taking an exception to IRQ mode, and copied to PSTATE.F on executing an illegal exception return operation in IRQ mode.

The reset behavior of this field is:

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

T, bit [5]

T32 Instruction set state. Set to the value of PSTATE.T on taking an exception to IRQ mode, and copied to PSTATE.T on executing an illegal exception return operation in IRQ mode.

The reset behavior of this field is:

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

M[4:0], bits [4:0]

Mode. Set to the value of PSTATE.M[4:0] on taking an exception to IRQ mode, and copied to PSTATE.M[4:0] on executing an illegal exception return operation in IRQ mode.

M[4:0]	Meaning
`0b10000`	User.
`0b10001`	FIQ.
`0b10010`	IRQ.
`0b10011`	Supervisor.
`0b10111`	Abort.
`0b11011`	Undefined.
`0b11111`	System.

Other values are reserved. If SPSR_irq.M[4:0] has a Reserved value, or a value for an unimplemented Exception level, executing an illegal exception return operation in IRQ mode is an illegal return event, as described in 'Illegal return events from AArch32 state'.

The reset behavior of this field is:

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

Accessing SPSR_irq

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

MRS <Xt>, SPSR_irq

op0	op1	CRn	CRm	op2
0b11	0b100	0b0100	0b0011	0b000

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.NV == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    X[t, 64] = SPSR_irq;
elsif PSTATE.EL == EL3 then
    X[t, 64] = SPSR_irq;
```

MSR SPSR_irq, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b100	0b0100	0b0011	0b000

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.NV == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    SPSR_irq = X[t, 64];
elsif PSTATE.EL == EL3 then
    SPSR_irq = X[t, 64];
```

AArch32 Registers | AArch64 Registers | AArch32 Instructions | AArch64 Instructions | Index by Encoding | External Registers

28/03/2023 16:01; 72747e43966d6b97dcbd230a1b3f0421d1ea3d94