ID_DFR0_EL1, AArch32 Debug Feature Register 0

The ID_DFR0_EL1 characteristics are:

Purpose

Provides top level information about the debug system in AArch32 state.

Must be interpreted with the Main ID Register, MIDR_EL1.

For general information about the interpretation of the ID registers see 'Principles of the ID scheme for fields in ID registers'.

Configuration

AArch64 System register ID_DFR0_EL1 bits [31:0] are architecturally mapped to AArch32 System register ID_DFR0[31:0].

Attributes

ID_DFR0_EL1 is a 64-bit register.

Field descriptions

When AArch32 is supported:

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							
TraceFilt	PerfMon	MProfDbg	MMapTrc	CopTrc	MMapDbg	CopSDbg	CopDbg
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.

TraceFilt, bits [31:28]

Armv8.4 Self-hosted Trace Extension version. Defined values are:

TraceFilt	Meaning
0b0000	Armv8.4 Self-hosted Trace Extension not implemented.
0b0001	Armv8.4 Self-hosted Trace Extension implemented.

All other values are reserved.

FEAT_TRF implements the functionality added by the value `0b0001`.

From Armv8.3, the permitted values are `0b0000` and `0b0001`.

PerfMon, bits [27:24]

Performance Monitors Extension version.

This field does not follow the standard ID scheme, but uses the alternative ID scheme described in 'Alternative ID scheme used for the Performance Monitors Extension version'

Defined values are:

PerfMon	Meaning
`0b0000`	Performance Monitors Extension not implemented.
`0b0001`	Performance Monitors Extension, PMUv1 implemented.
`0b0010`	Performance Monitors Extension, PMUv2 implemented.
`0b0011`	Performance Monitors Extension, PMUv3 implemented.
`0b0100`	PMUv3 for Armv8.1. As `0b0011`, and adds support for: • Extended 16-bit PMEVTYPER<n>.evtCount field. • If EL2 is implemented, the HDCR.HPMD control.
`0b0101`	PMUv3 for Armv8.4. As `0b0100`, and adds support for the PMMIR register.
`0b0110`	PMUv3 for Armv8.5. As `0b0101`, and adds support for: • 64-bit event counters. • If EL2 is implemented, the HDCR.HCCD control. • If EL3 is implemented, the MDCR_EL3.SCCD control.

0b0111	PMUv3 for Armv8.7. As 0b0110, and adds support for: • The PMCR.FZO and, if EL2 is implemented, HDCR.HPMFZO controls. • If EL3 is implemented, the MDCR_EL3.{MPMX,MCCD} controls.
0b1000	PMUv3 for Armv8.8. As 0b0111, and: • Extends the Common event number space to include 0x0040 to 0x00BF and 0x4040 to 0x40BF. • Removes the constrained unpredictable behaviors if a reserved or unimplemented PMU event number is selected.
0b1001	PMUv3 for Armv8.9. As 0b1000, and: • Updates the definitions of existing PMU events. • Adds support for the EDECR.PME control.
0b1111	implementation defined form of performance monitors supported, PMUv3 not supported. Arm does not recommend this value for new implementations.

All other values are reserved.

FEAT_PMUv3 implements the functionality identified by the value 0b0011.

FEAT_PMUv3p1 implements the functionality identified by the value 0b0100.

FEAT_PMUv3p4 implements the functionality identified by the value 0b0101.

FEAT_PMUv3p5 implements the functionality identified by the value 0b0110.

FEAT_PMUv3p7 implements the functionality identified by the value 0b0111.

FEAT_PMUv3p8 implements the functionality identified by the value 0b1000.

FEAT_PMUv3p9 implements the functionality identified by the value 0b1001.

In any Armv8 implementation, the values 0b0001 and 0b0010 are not permitted.

From Armv8.1, if FEAT_PMUv3 is implemented, the value 0b0011 is not permitted.

From Armv8.4, if FEAT_PMUv3 is implemented, the value 0b0100 is not permitted.

From Armv8.5, if FEAT_PMUv3 is implemented, the value 0b0101 is not permitted.

From Armv8.7, if FEAT_PMUv3 is implemented, the value 0b0110 is not permitted.

From Armv8.8, if FEAT_PMUv3 is implemented, the value 0b0111 is not permitted.

From Armv8.9, if FEAT_PMUv3 is implemented, the value 0b1000 is not permitted.

MProfDbg, bits [23:20]

M-profile Debug. Support for memory-mapped debug model for M-profile processors. Defined values are:

MProfDbg	Meaning
0b0000	Not supported.
0b0001	Support for M-profile Debug architecture, with memory-mapped access.

All other values are reserved.

In Armv8-A, the only permitted value is 0b0000.

MMapTrc, bits [19:16]

Memory-mapped Trace. Support for memory-mapped trace model. Defined values are:

MMapTrc	Meaning

0b0000	Not supported.
0b0001	Support for Arm trace architecture, with memory-mapped access.

All other values are reserved.

In Armv8-A, the permitted values are 0b0000 and 0b0001.

For more information, see the ARMÂ® Embedded Trace Macrocell Architecture Specification, ETMv4 (ARM IHI 0064).

CopTrc, bits [15:12]

Support for System registers-based trace model, using registers in the coproc == 0b1110 encoding space. Defined values are:

CopTrc	Meaning
0b0000	Not supported.
0b0001	Support for Arm trace architecture, with System registers access.

All other values are reserved.

In Armv8-A, the permitted values are 0b0000 and 0b0001.

For more information, see the ARMÂ® Embedded Trace Macrocell Architecture Specification, ETMv4 (ARM IHI 0064).

MMapDbg, bits [11:8]

Memory-mapped Debug. Support for Armv7 memory-mapped debug model for A and R-profile processors. Defined values are:

MMapDbg	Meaning
0b0000	Not supported.
0b0100	Support for Armv7, v7 Debug architecture, with memory-mapped access.
0b0101	Support for Armv7, v7.1 Debug architecture, with memory-mapped access.

All other values are reserved.

In Armv8-A, the only permitted value is 0b0000.

The optional memory map defined by Armv8 is not compatible with Armv7.

CopSDbg, bits [7:4]

Support for a System registers-based Secure debug model, using registers in the coproc = `0b1110` encoding space, for an A-profile processor that includes EL3.

If EL3 is not implemented and the implemented Security state is Non-secure state, this field is res0. Otherwise, this field reads the same as bits [3:0].

CopDbg, bits [3:0]

Debug architecture version. Indicates presence of Armv8 debug architecture. Defined values are:

CopDbg	Meaning
`0b0000`	Not supported.
`0b0010`	Armv6, v6 Debug architecture, with System registers access.
`0b0011`	Armv6, v6.1 Debug architecture, with System registers access.
`0b0100`	Armv7, v7 Debug architecture, with System registers access.
`0b0101`	Armv7, v7.1 Debug architecture, with System registers access.
`0b0110`	Armv8 debug architecture.
`0b0111`	Armv8 debug architecture with Virtualization Host Extensions.
`0b1000`	Armv8.2 debug architecture, FEAT_Debugv8p2.
`0b1001`	Armv8.4 debug architecture, FEAT_Debugv8p4.
`0b1010`	Armv8.8 debug architecture, FEAT_Debugv8p8.
`0b1011`	Armv8.9 debug architecture, FEAT_Debugv8p9.

All other values are reserved.

The values `0b0000`, `0b0010`, `0b0011`, `0b0100`, and `0b0101` are not permitted in Armv8.

FEAT_VHE adds the functionality identified by the value `0b0111`.

FEAT_Debugv8p2 adds the functionality identified by the value `0b1000`.

FEAT_Debugv8p4 adds the functionality identified by the value `0b1001`.

FEAT_Debugv8p8 adds the functionality identified by the value `0b1010`.

FEAT_Debugv8p9 adds the functionality identified by the value `0b1011`.

From Armv8.1, when FEAT_VHE is implemented the value `0b0110` is not permitted.

From Armv8.2, the values `0b0110` and `0b0111` are not permitted.

From Armv8.4, the value `0b1000` is not permitted.

From Armv8.8, the value `0b1001` is not permitted.

From Armv8.9, the value `0b1010` is not permitted.

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
UNKNOWN																															
UNKNOWN																															
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, unknown.

Accessing ID_DFR0_EL1

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

MRS <Xt>, ID_DFR0_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0000	0b0001	0b010

```
if PSTATE.EL == EL0 then
    if IsFeatureImplemented(FEAT_IDST) then
        if EL2Enabled() && HCR_EL2.TGE == '1' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        else
            AArch64.SystemAccessTrap(EL1, 0x18);
    else
        UNDEFINED;
```

```
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TID3 == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = ID_DFR0_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = ID_DFR0_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = ID_DFR0_EL1;
```

28/03/2023 16:02; 72747e43966d6b97dcbd230a1b3f0421d1ea3d94