## NAME

**dwarf_get_fde_info_for_reg3 -** retrieve register rule

## LIBRARY

DWARF Access Library (libdwarf, -ldwarf)

## SYNOPSIS

**#include <libdwarf.h>**

*int*
**dwarf_get_fde_info_for_reg3**(*Dwarf_Fde fde*, *Dwarf_Half table_column*, *Dwarf_Addr pc*,
*Dwarf_Small *type*, *Dwarf_Signed *offset_relevant*, *Dwarf_Signed *register_num*,
*Dwarf_Signed *offset_or_block_len*, *Dwarf_Ptr *block_ptr*, *Dwarf_Addr *row_pc*,
*Dwarf_Error *error*);

## DESCRIPTION

Function **dwarf_get_fde_info_for_reg3**() retrieves a register rule from the register rule table associated with a given FDE descriptor, given a program counter address and rule column number.

Argument *fde* should reference a valid DWARF FDE descriptor.

Arugment *table_column* should hold the column number of the register rule desired.

Argument *pc* should hold the program counter address to be used to locate the desired register rule row.

On successful execution, **dwarf_get_fde_info_for_reg3**() stores information about the register rule found into the locations pointed to by the arguments *type*, *offset_relevant*, *register_num*, *offset_or_block_len*, *block_ptr* and *row_pc*.

Argument *type* should point to a location which will hold the type code of the register rule found. The returned value is one of the DW_EXPR_* contants defined in the header file <*libdwarf.h*>.

If there is an offset value associated with the register rule, the location pointed to by argument *offset_relevant* will be set to 1.

Argument *register_num* should point to a location which will hold the register number associated with the register rule.

If the register rule is of type DW_EXPR_OFFSET or DW_EXPR_VAL_OFFSET, the location pointed to by argument *offset_or_block_len* will be set to the offset value associated with the register rule, or to

0 if the register rule does not have an offset value. If the type code is DW_EXPR_EXPRESSION or DW_EXPR_VAL_EXPRESSION, the location pointed to by argument *offset_or_block_len* will be set to the length in bytes of the DWARF expression block associated with the register rule.

Argument *block_ptr* should point to a location which will be set to a pointer to the content of the DWARF expression block associated with the register rule.

Argument *row_pc* should point to a location which will be set to the lowest program counter address associated with the register rule found.

If argument *err* is not NULL, it will be used to return an error descriptor in case of an error.

## RETURN VALUES

Function **dwarf_get_fde_info_for_reg3**() returns DW_DLV_OK when it succeeds. In case of an error, it returns DW_DLV_ERROR and sets the argument *err*.

## EXAMPLES

To retrieve the register rules at column 3 from a rule table associated with a FDE descriptor:

```
Dwarf_Fde fde;
Dwarf_Off fde_offset, cie_offset;
Dwarf_Unsigned func_len, fde_length;
Dwarf_Signed cie_index, offset_relevant, register_num;
Dwarf_Signed offset_or_block_len;
Dwarf_Addr low_pc, row_pc;
Dwarf_Ptr fde_addr, block_ptr;
Dwarf_Small type;
Dwarf_Error de;

/* ... assuming ‘fde‘ references a valid FDE descriptor... */
if (dwarf_get_fde_range(fde, &low_pc, &func_len, &fde_addr,
    &fde_length, &cie_offset, &cie_index, &fde_offset,
    &de) != DW_DLV_OK)
        errx(EXIT_FAILURE, "dwarf_get_fde_range failed: %s",
            dwarf_errmsg(de));

/* Iterate all the table rows. */
for (pc = low_pc; pc < low_pc + func_len; pc++) {
        if (dwarf_get_fde_info_for_reg3(fde, 3, pc, &type,
            &offset_relevant, &register_num, &offset_or_block_len,
            &block_ptr, &row_pc, &de) != DW_DLV_OK) {
                warnx("dwarf_get_fde_info_for_reg3 failed: %s",
                    dwarf_errmsg(de));
                continue;
        }
        /* ... use the retrieved register rule ... */
}
```

## ERRORS

Function **dwarf_get_fde_info_for_reg3**() can fail with:

[DW_DLE_ARGUMENT] One of the arguments *block_ptr*, *fde*, *offset_or_block_len*, *offset_relevant*, *register_num*, *row_pc*, or *type* was NULL.

[DW_DLE_FRAME_TABLE_COL_BAD] The column number provided in argument *table_column* was too large.

[DW_DLE_PC_NOT_IN_FDE_RANGE] The program counter value provided in argument *pc* did not fall in the range covered by argument *fde*.

## SEE ALSO

dwarf(3), dwarf_get_fde_at_pc(3), dwarf_get_fde_info_for_all_regs(3), dwarf_get_fde_info_for_all_regs3(3), dwarf_get_fde_info_for_cfa_reg3(3), dwarf_get_fde_info_for_reg(3), dwarf_get_fde_n(3), dwarf_set_frame_cfa_value(3), dwarf_set_frame_rule_initial_value(3), dwarf_set_frame_rule_table_size(3), dwarf_set_frame_same_value(3), dwarf_set_frame_undefined_value(3)