NAME

s390_runtime_instr − enable/disable s390 CPU run-time instrumentation

SYNOPSIS

#include <asm/runtime_instr.h>

int s390_runtime_instr(int *command***, int** *signum***);**

DESCRIPTION

The **s390_runtime_instr**() system call starts or stops CPU run-time instrumentation for the calling thread.

The *command* argument controls whether run-time instrumentation is started (**S390_RUNTIME_IN-STR_START**, 1) or stopped (**S390_RUNTIME_INSTR_STOP**, 2) for the calling thread.

The *signum* argument specifies the number of a real-time signal. The real-time signal is sent to the thread if the run-time instrumentation buffer is full or if the run-time-instrumentation-halted interrupt occurred.

RETURN VALUE

On success, **s390_runtime_instr**() returns 0 and enables the thread for run-time instrumentation by assigning the thread a default run-time instrumentation control block. The caller can then read and modify the control block and start the run-time instrumentation. On error, −1 is returned and *errno* is set to one of the error codes listed below.

ERRORS

EINVAL
The value specified in *command* is not a valid command or the value specified in *signum* is not a real-time signal number.

ENOMEM
Allocating memory for the run-time instrumentation control block failed.

EOPNOTSUPP
The run-time instrumentation facility is not available.

VERSIONS

This system call is available since Linux 3.7.

CONFORMING TO

This Linux-specific system call is available only on the s390 architecture. The run-time instrumentation facility is available beginning with System z EC12.

NOTES

Glibc does not provide a wrapper for this system call, use **syscall**(2) to call it.

The *asm/runtime_instr.h* header file is available since Linux 4.16.

SEE ALSO

syscall(2), **signal**(7)

COLOPHON

This page is part of release 5.02 of the Linux *man-pages* project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at https://www.kernel.org/doc/man−pages/.