NAME

intspy - Monitor interrupts while running a program.

SYNOPSIS

intspy [**-**setup] logfile command [parameters]

DESCRIPTION

Intspy runs command with optional parameters, logging interrupts to logfile. Which interrupts have to be monitored, and what has to be logged is defined in the file **intspy.ini**. This file is searched for, in the current directory. Next all directories included in the **PATH** environment variable are searched. The format of **intspy.ini** is the standard Windows ini file format.

The interrupts to be monitored are specified in the section **[intspy.**setup**]**, or in **[intspy.default]** if setup is not specified at the command line. The lines in **[intspy.**setup**]** or **[intspy.default]** look like:

intno **=** descriptionsection

Intno is a hexadecimal number defining the interrupt to be monitored. Descriptionsection is a string defining the section in which pairs of masks and descriptions are defined. Descriptionsection defaults to **default**.

If setup is a hexadecimal integer, the contents of **[intspy.**setup**]** default to:

setup **= default**

Otherwise an error message is generated when **[intspy.**setup**]** is empty.

The lines in **[**descriptionsection**]** or **[default]** look like:

mask **=** format [**,** var1 [**,** var2 ...]]

If interrupt intno occurs, mask is compared with the contents of the registers AH, AL, BH, BL, CH, CL, DH and DL. If mask matches the register contents, a message defined by format, var1, var2, etc. is written to logfile. Only the message defined by the first matching mask is written. The first matching mask is the one which occurs first in the **intspy.ini** file.

Mask contains 1 to 8 pairs of hexadecimal digits or the wildcard **??**. The pairs are compared with the registers in the following order, AH, AL, BH, BL, CH, CL, DH and DL. Missing pairs are assumed to be wildcards.

Format is a double quoted string confirming to the syntax of the C programming language, containing conversion specifications like the format string of **printf()**.

Var1, var2, etc. are pseudo variables. Next is a list of possible pseudo variables, together with the values they represent:

ax	The contents of the AX register when the interrupt was generated.
ah	The contents of the AH register when the interrupt was generated.
al	The contents of the AL register when the interrupt was generated.
bx	The contents of the BX register when the interrupt was generated.
bh	The contents of the BH register when the interrupt was generated.
bl	The contents of the BL register when the interrupt was generated.
cx	The contents of the CX register when the interrupt was generated.
ch	The contents of the CH register when the interrupt was generated.
cl	The contents of the CL register when the interrupt was generated.
dx	The contents of the DX register when the interrupt was generated.
dh	The contents of the DH register when the interrupt was generated.
dl	The contents of the DL register when the interrupt was generated.
ds	The contents of the DS register when the interrupt was generated.
es	The contents of the ES register when the interrupt was generated.
cs	The contents of the CS register when the interrupt was generated.
ss	The contents of the SS register when the interrupt was generated.
di	The contents of the DI register when the interrupt was generated.
si	The contents of the SI register when the interrupt was generated.
bp	The contents of the BP register when the interrupt was generated.
sp	The contents of the SP register when the interrupt was generated.
ip	The contents of the IP register when the interrupt was generated.
flags	The contents of the flags register when the interrupt was generated.
dsax	A pointer to a copy of the string pointed to by DS:AX. This copy is truncated, and all non printable characters are replaced.
dsbx	A pointer to a copy of the string pointed to by DS:BX. This copy is truncated, and all non printable characters are replaced.
dscx	A pointer to a copy of the string pointed to by DS:CX. This copy is truncated, and all non printable characters are replaced.
dsdx	A pointer to a copy of the string pointed to by DS:DX. This copy is truncated, and all non printable characters are replaced.
dsdi	A pointer to a copy of the string pointed to by DS:DI. This copy is truncated, and all non printable characters are replaced.
dssi	A pointer to a copy of the string pointed to by DS:SI. This copy is truncated, and all non printable characters are replaced.
esax	A pointer to a copy of the string pointed to by ES:AX. This copy is truncated, and all non printable characters are replaced.
esbx	A pointer to a copy of the string pointed to by ES:BX. This copy is truncated, and all non printable characters are replaced.
escx	A pointer to a copy of the string pointed to by ES:CX. This copy is truncated, and all non printable characters are replaced.
esdx	A pointer to a copy of the string pointed to by ES:DX. This copy is

truncated, and all non printable characters are replaced.

esdi A pointer to a copy of the string pointed to by ES:DI. This copy is truncated, and all non printable characters are replaced.

essi A pointer to a copy of the string pointed to by ES:SI. This copy is truncated, and all non printable characters are replaced.

csax A pointer to a copy of the string pointed to by CS:AX. This copy is truncated, and all non printable characters are replaced.

csbx A pointer to a copy of the string pointed to by CS:BX. This copy is truncated, and all non printable characters are replaced.

cscx A pointer to a copy of the string pointed to by CS:CX. This copy is truncated, and all non printable characters are replaced.

csdx A pointer to a copy of the string pointed to by CS:DX. This copy is truncated, and all non printable characters are replaced.

csdi A pointer to a copy of the string pointed to by CS:DI. This copy is truncated, and all non printable characters are replaced.

cssi A pointer to a copy of the string pointed to by CS:SI. This copy is truncated, and all non printable characters are replaced.

ssax A pointer to a copy of the string pointed to by SS:AX. This copy is truncated, and all non printable characters are replaced.

ssbx A pointer to a copy of the string pointed to by SS:BX. This copy is truncated, and all non printable characters are replaced.

sscx A pointer to a copy of the string pointed to by SS:CX. This copy is truncated, and all non printable characters are replaced.

ssdx A pointer to a copy of the string pointed to by SS:DX. This copy is truncated, and all non printable characters are replaced.

ssdi A pointer to a copy of the string pointed to by SS:DI. This copy is truncated, and all non printable characters are replaced.

sssi A pointer to a copy of the string pointed to by SS:SI. This copy is truncated, and all non printable characters are replaced.

intno
The number of the interrupt.

BUGS

For interrupt handlers, it is at some instances not allowed to write to a file. In this case logging messages are buffered, and written upon a later interrupt. If the buffer overflows, all messages in the buffer are discarded, and an error message is written to the logging file. If this happens, it is a good idea to monitor also an interrupt which occurs often, and have an empty string to be written on this interrupt. This causes the buffer to be flushed regularly.

Intspy uses the C function **spawnvp()** to execute <u>command</u>. The interrupts generated by this call are logged too.

When running a program in a Windows DOS box, some interrupts are eaten by Windows, and never reach **intspy**.

Some programs install their own interrupt handler. This could mean that the interrupt monitor of **intspy** is replaced by the programs interrupt handler, so nothing is logged.

Some interrupt vectors are no real interrupt vectors, but are used as a storage location for something. Installing an interrupt monitor for the corresponding interrupt may cause unexpected results.

VERSION

1.0

COPYRIGHT