NAME

efirt, **efi_rt_ok**, **efi_get_table**, **efi_get_time**, **efi_get_time_capabilities**, **efi_reset_system**, **efi_set_time**, **efi_var_get**, **efi_var_nextname**, **efi_var_set** - kernel access to UEFI runtime services

SYNOPSIS

options EFIRT

#include <sys/efi.h>

int
efi_rt_ok(*void*);

int
efi_get_table(*struct uuid *uuid*, *void **ptr*);

int
efi_get_time(*struct efi_tm *tm*);

int
efi_get_time_capabilities(*struct efi_tmcap *tmcap*);

int
efi_reset_system(*enum efi_reset type*);

int
efi_set_time(*struct efi_tm *tm*);

int
efi_var_get(*uint16_t *name*, *struct uuid *vendor*, *uint32_t *attrib*, *size_t *datasize*, *void *data*);

int
efi_var_nextname(*size_t *namesize*, *uint16_t *name*, *struct uuid *vendor*);

int
efi_var_set(*uint16_t *name*, *struct uuid *vendor*, *uint32_t attrib*, *size_t datasize*, *void *data*);

DESCRIPTION

All of the following calls will return ENXIO if UEFI runtime services are not available. **efirt** is currently only available on amd64 and arm64.

The **efi_rt_ok**() Returns 0 if UEFI runtime services are present and functional, or ENXIO if not.

The **efi_get_table**() function gets a table by uuid from the UEFI system table. Returns 0 if the table was found and populates *ptr with the address. Returns ENXIO if the configuration table or system table are unset. Returns ENOENT if the requested table cannot be found.

The **efi_get_time**() function gets the current time from the RTC, if available. Returns 0 and populates the *struct efi_tm* on success. Returns EINVAL if the *struct efi_tm* is NULL, or EIO if the time could not be retrieved due to a hardware error.

The **efi_get_time_capabilities**() function gets the capabilities from the RTC. Returns 0 and populates the *struct efi_tmcap* on success. Returns EINVAL if the *struct efi_tm* is NULL, or EIO if the time could not be retrieved due to a hardware error.

The **efi_reset_system**() function requests a reset of the system. The *type* argument may be one of the *enum efi_reset* values:

EFI_RESET_COLD Perform a cold reset of the system, and reboot.

EFI_RESET_WARM Perform a warm reset of the system, and reboot.

EFI_RESET_SHUTDOWN Power off the system.

The **efi_set_time**() function sets the time on the RTC to the time described by the *struct efi_tm* passed in. Returns 0 on success, EINVAL if a time field is out of range, or EIO if the time could not be set due to a hardware error.

The **efi_var_get**() function fetches the variable identified by *vendor* and *name*. Returns 0 and populates *attrib*, *datasize*, and *data* on success. Otherwise, one of the following errors are returned:

ENOENT The variable was not found.

EOVERFLOW *datasize* is not sufficient to hold the variable data. *namesize* is updated to reflect the size needed to complete the request.

EINVAL One of *name*, *vendor*, or *datasize* are NULL. Alternatively, *datasize* is large enough to hold the response but *data* is NULL.

EIO The variable could not be retrieved due to a hardware error.

EDOOFUS The variable could not be retireved due to an authentication failure.

The **efi_var_nextname**() function is used for enumeration of variables. On the initial call to **efi_var_nextname**(), *name* should be an empty string. Subsequent calls should pass in the last *name* and *vendor* returned until ENOENT is returned. Returns 0 and populates *namesize*, *name*, and *vendor* with the next variable's data. Otherwise, returns one of the following errors:

ENOENT The next variable was not found.

EOVERFLOW *datasize* is not sufficient to hold the variable data. *namesize* is updated to reflect the size needed to complete the request.

EINVAL One of *name*, *vendor*, or *datasize* are NULL.

EIO The variable could not be retrieved due to a hardware error.

The **efi_var_set**() function sets the variable described by *name* and *vendor*. Returns 0 if the variable has been successfully. Otherwise, returns one of the following errors:

EINVAL Either *attrib* was an invalid combination of attributes, *datasize* exceeds the maximum allowed size, or *name* is an empty Unicode stirng.

EAGAIN Not enough storage is available to hold the variable and its data.

EIO The variable could not be saved due to a hardware error.

EROFS The variable in question is read-only or may not be deleted.

EDOOFUS The variable could not be set due to an authentication failure.

ENOENT The variable trying to be updated or deleted was not found.

SEE ALSO

efidev(4)

AUTHORS

This manual page was written by Kyle Evans <*kevans@FreeBSD.org*>.