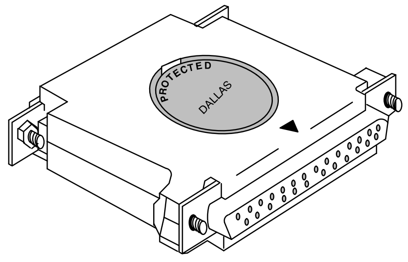
DALLAS SEMICONDUCTOR

DS1412 Serial Port Adapter

FEATURES

- RS232 interface for Dallas iButtons
- No external power required
- API included
- No gender changers needed
- Multiplatform Support



DESCRIPTION

The DS1412 Serial Port Adapter interfaces Dallas Semiconductor Authorization iButtons to host computers via an RS232 serial port. In conjunction with the iButtons, the DS1412 provides a high security storage vault for critical license management and execution control information. Only users that possess an iButton can utilize the software, preventing execution of unauthorized copies.

The modularity of the DS1412 allows for easy feature customization. The device supports the insertion of one iButton, which can be removed and replaced with another iButton of different functionality.

For example, a DS1427 Time iButton can be programmed for 30 day expiration, issued with a DS1412, and an evaluation software copy. Upon expiration, the evaluator can be converted into a registered user simply by issuing a DS1425 Multi iButton.

The DS1412 supports the same iButtons as other Dallas port adapters. This allows standardization of any protection scheme across virtually all hardware platforms, regardless of the operating system. The iButtons remain constant, and the port adapters change according to the specific platform interface.

DS1412 SOFTWARE

The DS1412 Development Kit contains an object file which must be linked with the application software in order to complete integration. This object file contains the low level interface for communicating with the DS1412.

An Application Programming Interface (API) has been provided as part of the DS1412 development software so that interfacing to the iButtons can be accomplished using high level commands.

The microprocessor contained in the DS1412 enables standard RS232 communication. The communication with the DS1412 is performed through a standard serial port device driver, eliminating the need to remake the UNIX kernel to install additional device drivers.

DS1412 HARDWARE

The DS1412 will integrate without modification onto different host computers, especially where their is no gender uniformity. One end of the DS1412 is a male connector and the other a female. Choose the proper end accordingly, connect to the serial port of the host, and the DS1412 will operate properly.

NOTE: The DS1412 requires a dedicated serial port regardless of which end is connected to the host.

A red label has been provided which should be affixed to the end the DS1412 not in use. This will ensure that no one attempts to connect another serial port device (a modem for example) in line with the DS1412.

DS1412 APPLICATION PROGRAM INTERFACE

DS1412api.c is provided to construct the DS1425 Software Key and DS1427 Time Key specific command structures and data packets which will be sent to the serial port transfer layer called sportio (serial port I/O). Note: The DS1425 and DS1427 iButtons are described in detail in their respective data sheets. The *take_census function is available for the DS1420, DS1425 and DS1427 iButtons. The following function is provided in DS1412api.c.

uchar *take_census (uchar *num_pres)

This function reads the ROM Data from all of the DOW (Dallas 1–wire) parts present on the 1–wire bus of the DS1412. take_census returns a pointer to the beginning of the buffer containing the ROM Data. num_pres indicates the number of DOW parts found during the census. If num_pres == 0 the structure DS1412_comm (described below) should be examined to determine the cause of the failure.

Please note that for many of the functions described below, a true result only implies that the data you specified was sent the proper iButton. For example, if read_subkey is called with an incorrect password in the key_dta array, the Multi iButton will send back incorrect secure data. However, read_subkey will return true (providing a DS1412 and iButton are present) as nobody but the caller has any way of knowing if the password and secure data are correct. To verify any data write or transfer the user can simply call the appropriate read function.

Please consult the iButton data sheets for details on their use.

DS1425 SPECIFIC FUNCTIONS

The following functions are available for communication with a DS1425 Multi iButton. For more information on the features of the iButton itself please refer to the DS1425 data sheet. Example programs and details of functions are provided in the documentation and example program provided with the API software.

uchar read_subkey (uchar key_num, uchar *key_dta, uchar *rom_ptr)

This function reads the secure subkey specified by key_num of the DS1425 with ROM Data pointed to by rom_ptr. key_dta points to a 64 byte field constructed as follows:

bytes 0–7	:	ID field
bytes 8–15	:	password field bytes
bytes 16–63	:	secure data field

Upon successful completion, read_subkey returns true, and the ID and secure data fields are overwritten with data from the key (the password field is left untouched as it is write only memory in the DS1425). If the function fails for any reason, read_subkey returns false. A structure defined below will specify the reason for the failure condition.

uchar write_subkey (uchar key_num, uchar *key_dta, uchar *rom_ptr)

This function writes to the secure subkey, specified by key_num, of the DS1425 with ROM Data pointed to by rom_ptr. The data sent to the part is taken from the field pointed to by key_dta. Upon successful completion, write_subkey returns true. If the function fails for any reason write_subkey returns false.

uchar read_1425_sp (uchar *scr_dta, uchar *rom_ptr)

This function reads the scratchpad of the DS1425 with ROM Data pointed to by rom_ptr. Upon successful completion read_1425_sp returns true with the 64 byte field pointed to byte scr_dta (this is simply a homogenous field of Read/ Write memory) overwritten with the scratchpad data from the selected DS1425. Any failure will result in read_1425_sp returning false.

uchar write_1425_sp (byte *scr_dta, byte *rom_ptr)

This function writes the scratchpad of the DS1425 with ROM Data pointed to by rom_ptr. The data sent to the part is taken from the field pointed to by scr_dta. Upon successful completion write_1425_sp returns true. Any failure will result in write_1425_sp returning false.

uchar reprogram_subkey (uchar key_num, uchar *key_dta, uchar *rom_ptr)

This function reprograms the subkey (specified by key_num) of the DS1425 with ROM Data pointed to by rom_ptr. The first 8 bytes of the key_dta array should contain the old ID field of the subkey to be reprogrammed. The second 8 bytes should contain the new ID, and the next eight bytes should hold your new password for the subkey. Upon successful completion reprogram_subkey returns true. Please note that a true result only implies the data was sent to the correct DS1425. To reprogram the subkey, the old ID field you pass this function must match that of the specified subkey in the DS1425. Any failure will result in reprogram_subkey returning false.

uchar move_block (uchar key_num, uchar *key_dta, uchar *rom_ptr, uchar blocknum)

This function performs the move block command of the DS1425 with ROM Data pointed to by rom_ptr. blocknum is a value from 0 to 8 which specifies the block to be transferred from the scratchpad, to the secure subkey number specified by key_num. The second 8 byte field of key_dta must contain the 8 byte password field which is required by the DS1425 for any block transfer. Upon successful completion move_block returns true. Please note that this does not imply that the move block occurred within the DS1425. If the password was incorrect the move block will be terminated by the DS1425. All successful completion guarantees is that the DS1425 received the specified data. Any failure will result in move_block returning false.

DS1427 SPECIFIC FUNCTIONS

The following functions are available for communication with a DS1427 Time iButton. For more information on the features of the iButton itself please refer to the DS1427 data sheet. Example programs and details of functions are provided in the documentation and example program provided with the API software.

uchar read_1427_smem (uchar *smem_dta, uchar *rom_ptr, uchar page_num)

This function reads the secure memory page (specified by page_num), of the DS1427 with ROM data pointed to by rom_ptr. smem_dta is a pointer to a 32 byte block of memory which will receive the secure memory data read from the DS1427. Upon successful completion read_1427_smem returns true, otherwise it returns false.

uchar read_1427_sp (uchar *scr_dta, uchar *rom_ptr)

This function reads the scratchpad of the DS1427 with ROM data pointed to by rom_ptr. The data read is stored beginning at the location pointed to by scr_dta. The first three bytes will receive the authorization code required for a copy scratchpad (described below). The next 32 bytes will be the scratchpad data just read. Upon successful completion read_1427_sp returns true, otherwise it returns false.

uchar write_1427_sp (uchar *scr_dta, uchar *rom_ptr, uchar page_num)

This function writes the scratchpad of the DS1427 with ROM data pointed to by rom_ptr. The data written to the scratchpad is taken starting with the byte pointed to by rom_ptr. page_num specifies which page you want this data copied to via a copy scratchpad command. Upon successful completion write_1427_sp returns true, otherwise it returns false.

uchar copy_1427_sp (uchar *auth_code, uchar *rom_ptr)

This function copies the contents of the scratchpad, with ROM Data pointed to by rom_ptr, to the secure memory starting at an address specified by the authorization code (first three bytes received from a read scratchpad). Remember that to set any of the write protect bits in the clock page (page #16), the copy scratchpad must be performed three times. After the first copy the AA bit is set. This means that the authorization code changes. The scratchpad can be read again, or you can set that bit in the authorization code to save time. Upon successful completion copy_1427_sp returns true, otherwise it returns false.

```
typedef struct
{
    char tty_name[20]
    uchar setup_error;
    uchar driver_open_error
    uchar ioctl_error
    uchar no_DS1412;
    uchar no_button;
    uchar mem_err;
}
DS1412_struct;

DS1412_struct DS1412_comm;
```

DS1412_comm: Contains fields used to examine error conditions and specify which tty driver is to be used to communicate with the DS1412.

tty_name: Copy the name of the tty driver for sportio to use for communication with the DS1412 in this field (i.e. /dev/ttya to use the serial port A on a SUN).

setup_error: This field is currently unused.

driver_open_error: If this field is true (1) tty_name is most likely invalid.

ioctl_error: If this field is true (1) an error occurred getting/setting the tty drivers parameters. This is not likely if tty_name is correct.

no_DS1412: This field is true (1) if no DS1412 was found on the serial port.

no_button: This field is true (1) if either no DS1412 was found, or if a DS1412 is present but contains no iButtons. If no_DS1412 is false (0), and no_button is true (1) prompt the user to insert an iButton.

mem_err: This field is true (1) if an error occurs attaching a shared memory segment. This is unlikely because sportio uses only a few bytes of the shared memory segment, and detaches the segment quickly.