Small Computer Systems Interface (SCSI) Toolbox for use in MATLAB™ (User's Guide)

Updated on: 24th October 2003
Version 2003.1.4

Introduction

The data storage industry in Singapore has gone through various stages of growth and we are widely regarded as the world's storage capital, playing host to major harddisk drives (HDDs) manufacturers such as IBM, Maxtor, Seagate, Matsushita-Kotobuki and others. As such, this industry will continue to be a vital part of the Singapore electronics cluster. Hence, it is essential for us to have a cheaper method of performing various testing functions like reading and writing of sector data to HDDs, for the purpose of failure analysis testing of HDDs. By using the software MATLAB™ to interface with the Parallel SCSI harddisk drive (HDD), we can effectively bypass the purchase and usage of other expensive commercial software.

This user guide introduces the basic MATLAB™ program written specifically to perform the functions of start, stop, read and write on a Parallel SCSI HDD. We describe the various dialogue boxes and explain how to use the program to perform the required test functions.

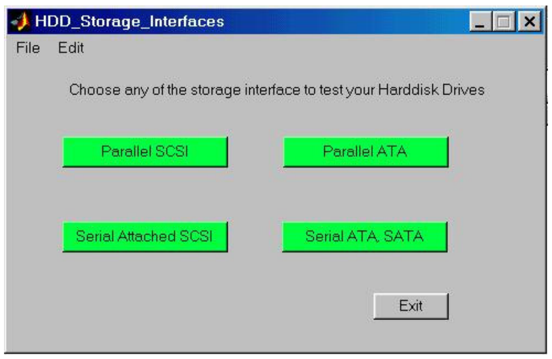
Hardware Used

SCSI ID #6: IBM IC35L018UWDY10-0
Ultrastar™ 320 SCSI Drive
Firmware: S32C
Capacity: 17501 Mbytes
RPM: 10000RPM
SCSI ID #7: Adaptec SCSI Card 29160
SCSI Terminator, Amphenal, Ultra-160

The program

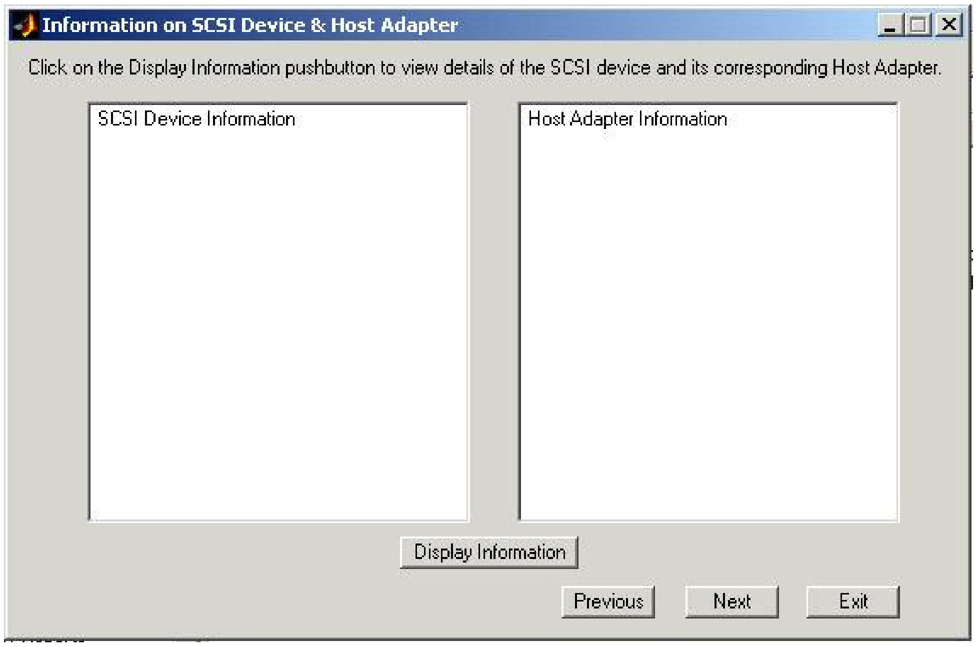
Firstly, go to the directory in which the relevant MATLAB M-files and other related files are stored. Right click on the file "HDD_Storage_Interfaces.m", and choose the 'Run' option from the menu. Upon starting the program, the following dialogue box will appear.

Step 1: Choosing your storage interface, and Display Information on SCSI Device & Host Adapter



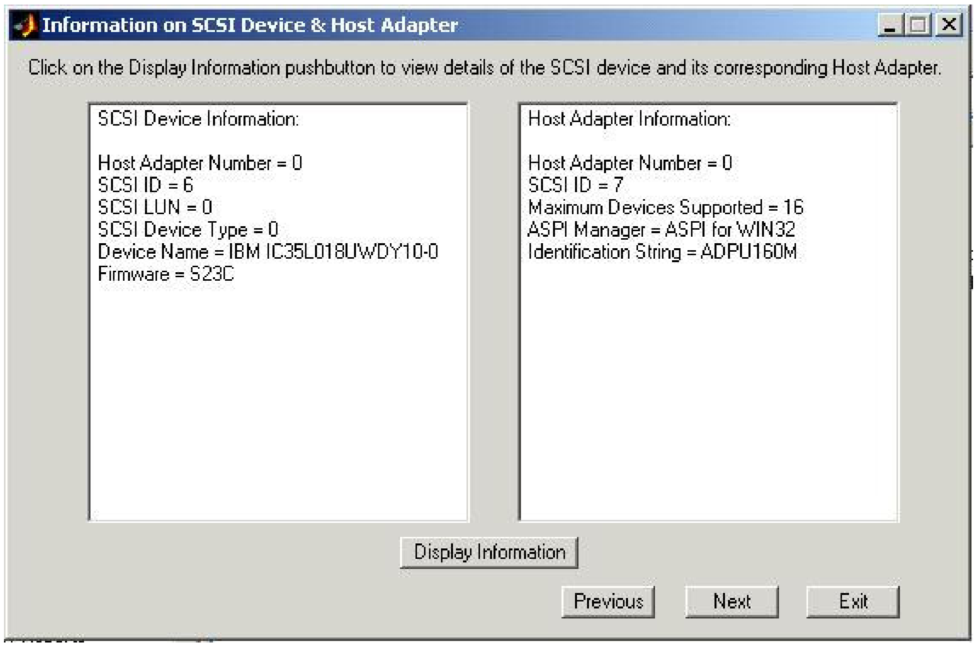
From the list of storage interfaces, the user can choose the appropriate interface for the HDD by clicking on the relevant option. However, our program currently only provides support for the Parallel SCSI storage interface. The possibilities of future developments on other storage interfaces like Serial Attached SCSI and Serial ATA are currently being looked into. Our focus is then on the Parallel SCSI storage interface.

After selecting the Parallel SCSI option, the following dialogue box will appear.



Click on the Display Information pushbutton to view details of the SCSI device and its corresponding Host Adapter.

Upon clicking on the Display Information pushbutton, the details of the SCSI Device and Host Adapter will be displayed as shown in the next dialog box.



Click on the Display Information pushbutton to view details of the SCSI device and its corresponding Host Adapter.

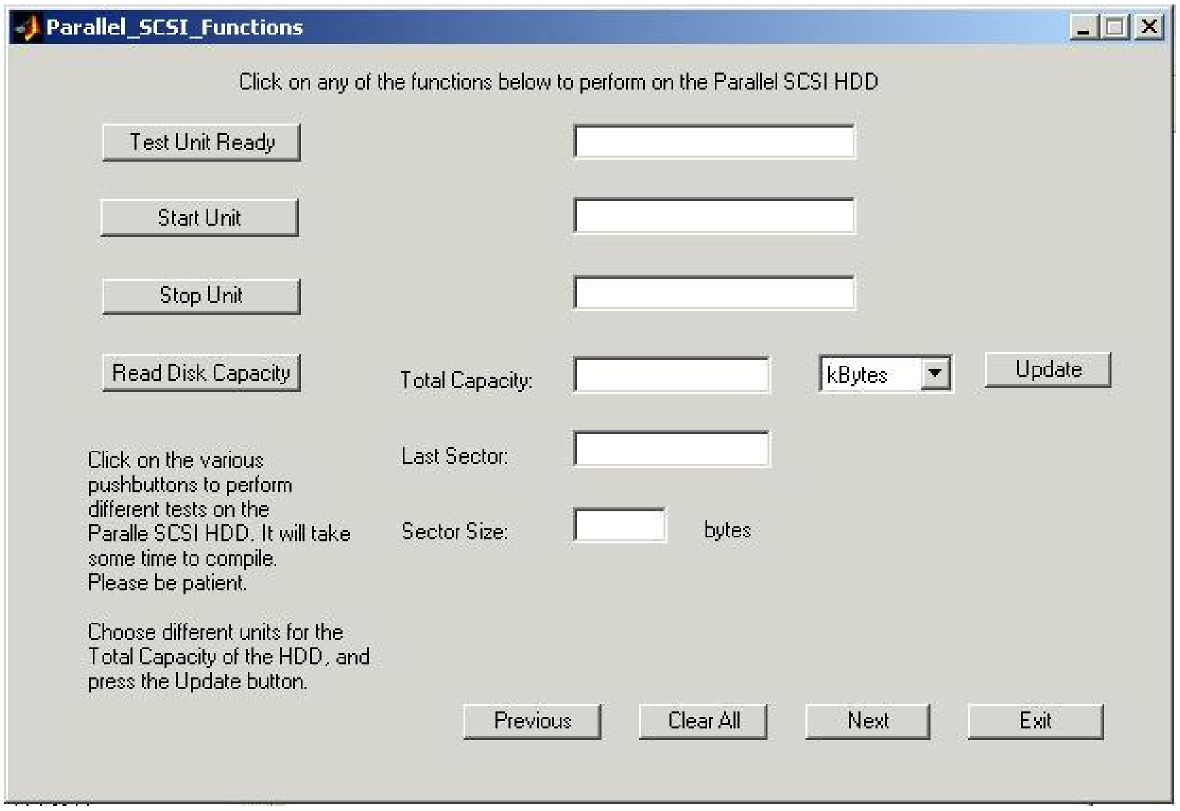
```
SCSI Device Information:

Host Adapter Number = 0
SCSI ID = 6
SCSI LUN = 0
SCSI Device Type = 0
Device Name = IBM IC35L018UWDY10-0
Firmware = S23C
```

```
Host Adapter Information:

Host Adapter Number = 0
SCSI ID = 7
Maximum Devices Supported = 16
ASPI Manager = ASPI for WIN32
Identification String = ADPU160M
```

Step 2: Performing functions of **Test Unit Ready**, **Start Unit**, **Stop Unit** and **Read Disk Capacity**.



Click on any of the functions below to perform on the Parallel SCSI HDD

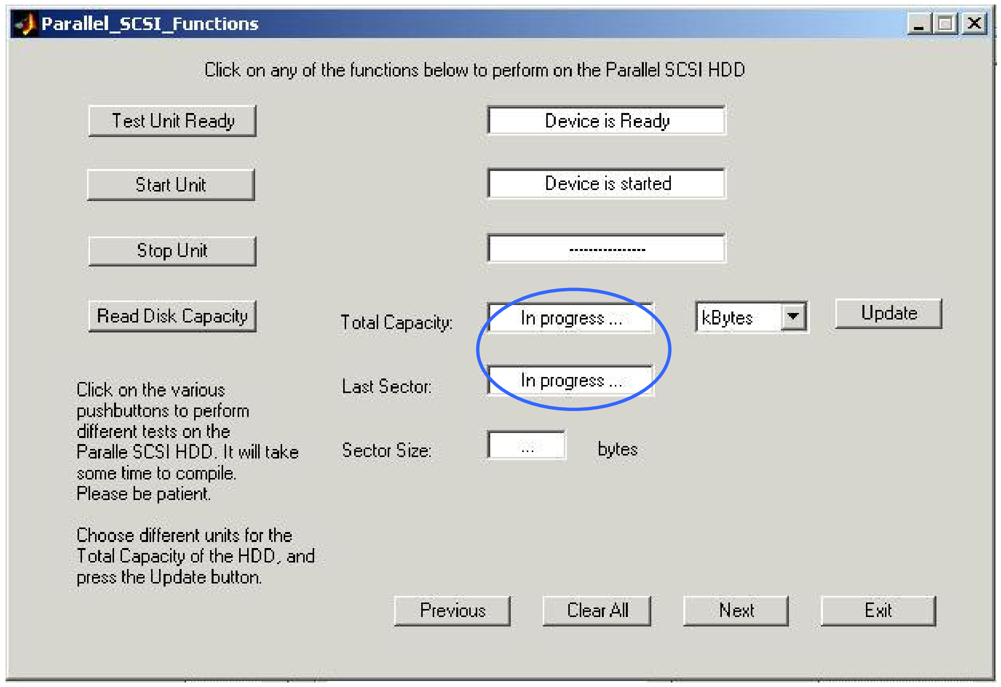
Click on the various pushbuttons to perform different tests on the Paralle SCSI HDD. It will take some time to compile. Please be patient.

Choose different units for the Total Capacity of the HDD, and press the Update button.

The above dialogue box enables the selection of various utilities for the Parallel SCSI HDD. Click on the various options to perform the corresponding function.

- *To test the device*, click on the 'Test Unit Ready' pushbutton. 'Device is Ready' message will be displayed on the edit box if the HDD is ready, else 'Device is NOT Ready' is shown.
- *To start the device*, click on the 'Start Unit' pushbutton. 'Device is started' message will be displayed on the edit box if the device has started.
- *To stop the device*, click on the 'Stop Unit' pushbutton. 'Device is stopped' message will be displayed on the edit box if the device has stopped.
- Note that the status of the device changes correspondingly in the 'Test Unit Ready' status box as the device is started or stopped.
- *To read the disk capacity of the device*, click on the 'Read Disk Capacity' pushbutton. Note that if the device has stopped, performing the Read Disk Capacity function will return a default value of 0 in the Total Capacity, Last Sector and Sector Size fields. We thus need to ensure that the device has started before performing the Read Disk Capacity function. The user can choose to display the Total Capacity either in kilobytes, megabytes, or gigabytes. Click the popup menu to choose the desired option and press the 'Update' pushbutton.

While performing the above functions, please wait patiently for a few seconds as the program needs to compile the necessary files. Note that in the process of executing the Read Disk Capacity function, the following will be displayed as an indication that the program is still running. (See next page)


Parallel_SCSI_Functions

Click on any of the functions below to perform on the Parallel SCSI HDD

Device is Ready

Device is started

Total Capacity: In progress ... kBytes

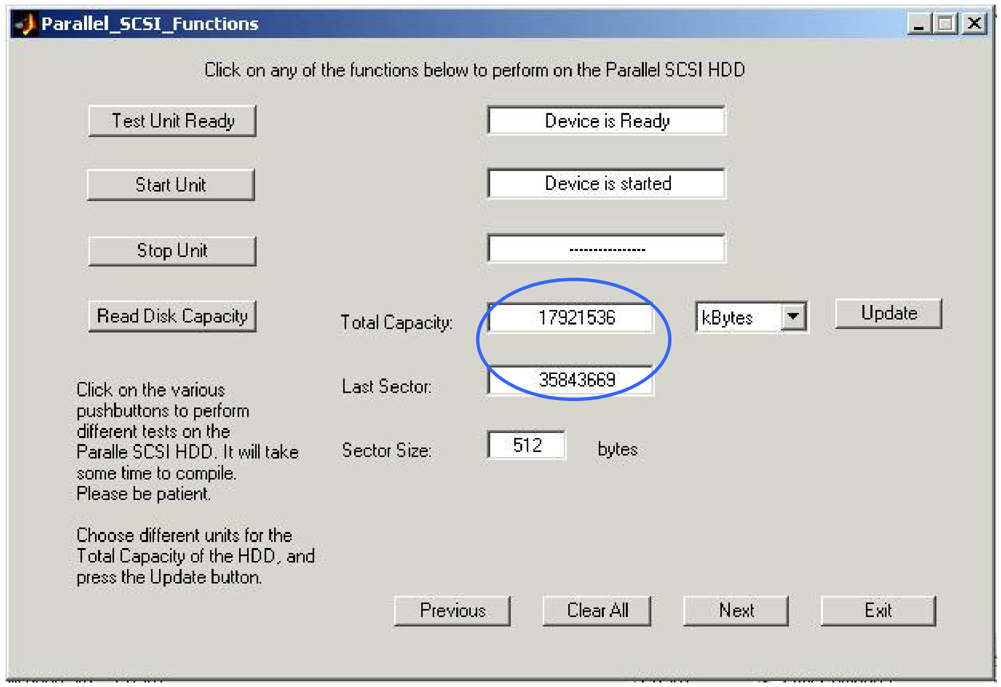
Last Sector: In progress ...

Sector Size: ... bytes

Click on the various pushbuttons to perform different tests on the Paralle SCSI HDD. It will take some time to compile. Please be patient.

Choose different units for the Total Capacity of the HDD, and press the Update button.

The dialogue box below illustrates what will be displayed after the program has performed the Read Disk Capacity function successfully.


Parallel_SCSI_Functions

Click on any of the functions below to perform on the Parallel SCSI HDD

Device is Ready

Device is started

Total Capacity: 17921536 kBytes

Last Sector: 35843669

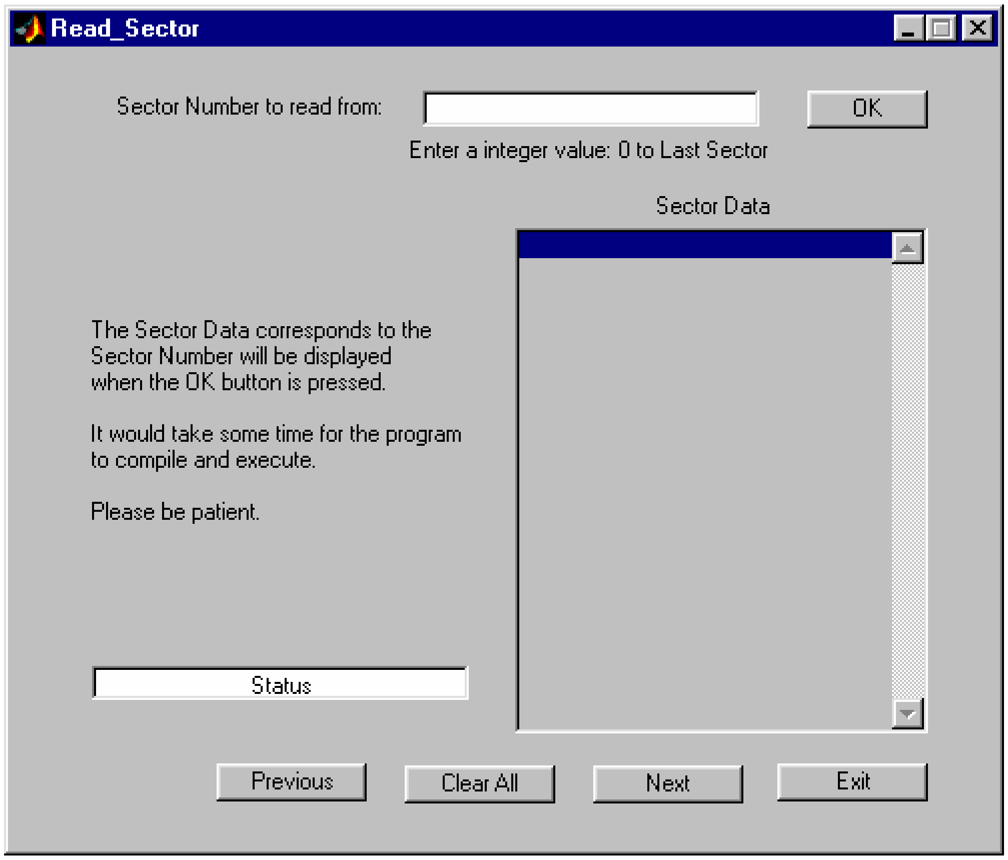
Sector Size: 512 bytes

Click on the various pushbuttons to perform different tests on the Paralle SCSI HDD. It will take some time to compile. Please be patient.

Choose different units for the Total Capacity of the HDD, and press the Update button.

To proceed to reading sector data, click on the ‘Next’ pushbutton.

Step 3: Performing the **Read Sector** function



Enter a integer value: 0 to Last Sector

The Sector Data corresponds to the Sector Number will be displayed when the OK button is pressed.

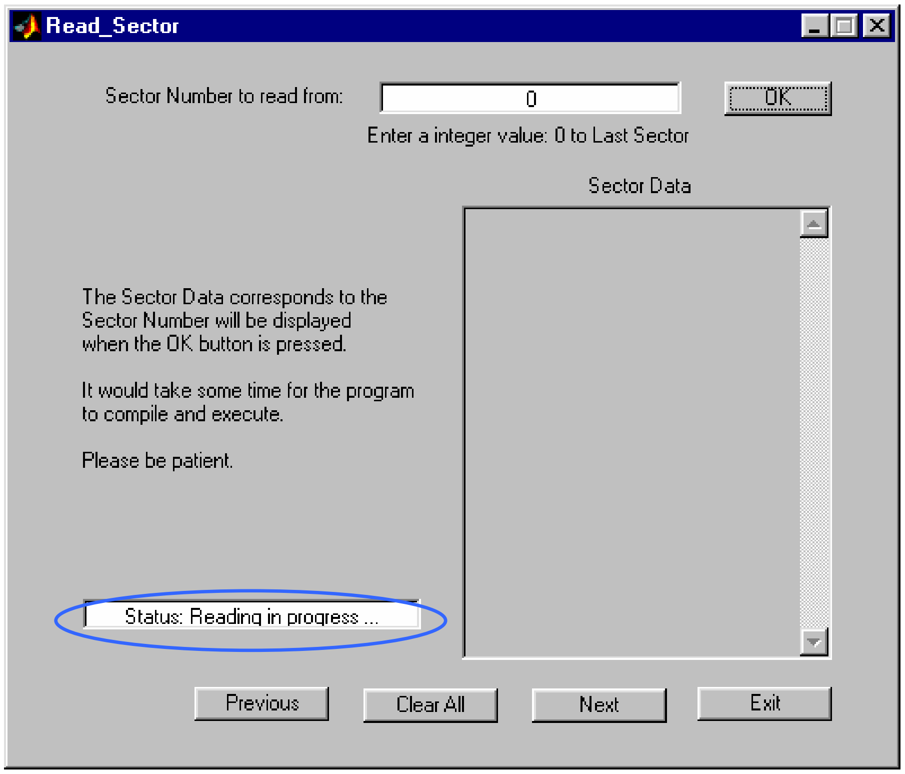
It would take some time for the program to compile and execute.

Please be patient.

To read the sector data of the HDD, the user has to input the required Sector Number in the edit box and then press the ‘OK’ pushbutton. The valid input can be any integer value from 0 to the Last Sector number.

The program will compile the necessary files and then read the data from the corresponding sector. These data will be displayed in the Sector Data text box. The status of the reading process is displayed in the edit box.

The two dialog boxes below illustrate what will be displayed when the reading process is carried out. Note that the illustration uses Sector Number 0 here.

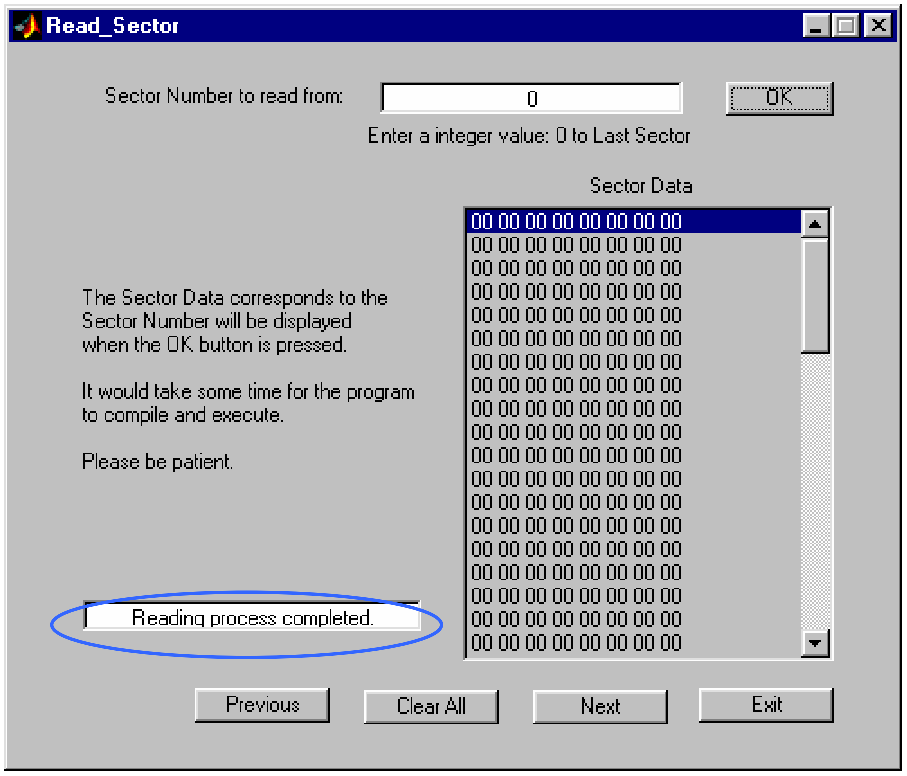

Read_Sector
Sector Number to read from:
0
OK
Enter a integer value: 0 to Last Sector
Sector Data
Previous
Clear All
Next
Exit

The Sector Data corresponds to the Sector Number will be displayed when the OK button is pressed.

It would take some time for the program to compile and execute.

Please be patient.

Status: Reading in progress ...


Read_Sector
Sector Number to read from:
0
OK
Enter a integer value: 0 to Last Sector
Sector Data
Previous
Clear All
Next
Exit

```
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
```

The Sector Data corresponds to the Sector Number will be displayed when the OK button is pressed.

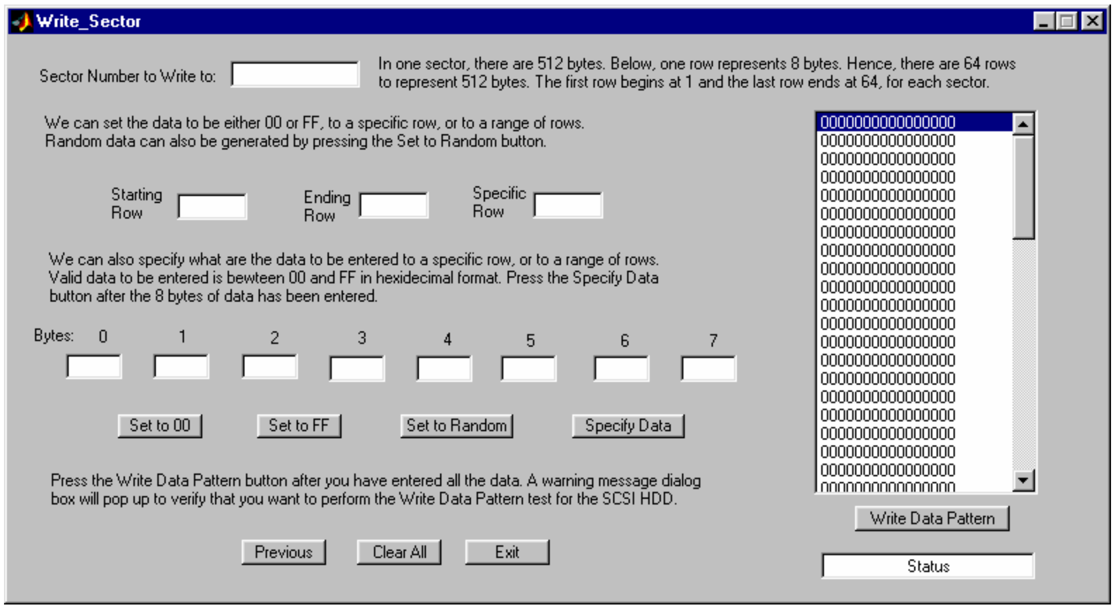
It would take some time for the program to compile and execute.

Please be patient.

Reading process completed.

To proceed with writing data into a particular sector, click on the ‘Next’ pushbutton.

Step 4: Performing the **Write Sector** function


Write_Sector

Sector Number to Write to:

In one sector, there are 512 bytes. Below, one row represents 8 bytes. Hence, there are 64 rows to represent 512 bytes. The first row begins at 1 and the last row ends at 64, for each sector.

We can set the data to be either 00 or FF, to a specific row, or to a range of rows. Random data can also be generated by pressing the Set to Random button.

Starting Row | Ending Row | Specific Row

We can also specify what are the data to be entered to a specific row, or to a range of rows. Valid data to be entered is bewteen 00 and FF in hexidecimal format. Press the Specify Data button after the 8 bytes of data has been entered.

Bytes: 0 1 2 3 4 5 6 7

Set to 00 | Set to FF | Set to Random | Specify Data

Press the Write Data Pattern button after you have entered all the data. A warning message dialog box will pop up to verify that you want to perform the Write Data Pattern test for the SCSI HDD.

Previous | Clear All | Exit

```
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
```

Write Data Pattern

Status

To write a data pattern to a particular sector, the user has to input the Sector Number of the HDD for the data to be written to. After that, input the specific row for the particular sector of the data. In a one typical sector, there are 512 bytes. In our program, one row of data represents 8 bytes. Hence, there are 64 rows to represent one sector of 512 bytes.

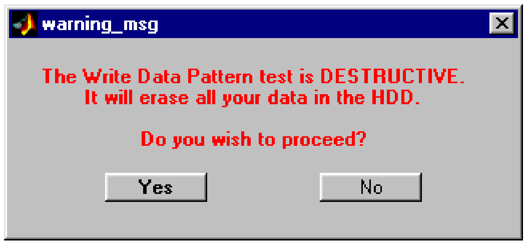
The user can either specify a range of rows (from 1 to 64) by inputting the 'Starting Row' and 'Ending Row' input boxes or specify a single row by keying the 'Specific Row' input box. After indicating the row numbers for that particular sector, the user can choose to write data in hexadecimal format of:

(1) '00', by pressing the 'Set to 00' pushbutton
(2) 'FF', by pressing the 'Set to FF' pushbutton
(3) Random data, by pressing the 'Set to Random' pushbutton
(4) User-defined data, by pressing the 'Specify Data' pushbutton

To specify what are the data to be written, simply fill in the user-defined data in the 8 edit boxes for the bytes 0 to 7.

The data pattern displayed on the right is the final data to be written to the particular sector. Click on the 'Write Data Pattern' pushbutton to start the writing process.

When the user clicks on the 'Write Data Pattern' pushbutton, a warning message dialogue box will appear. It warns that the writing process is destructive and will erase all the data in the HDD. Only when the user selects the 'Yes' pushbutton will the writing process continue.

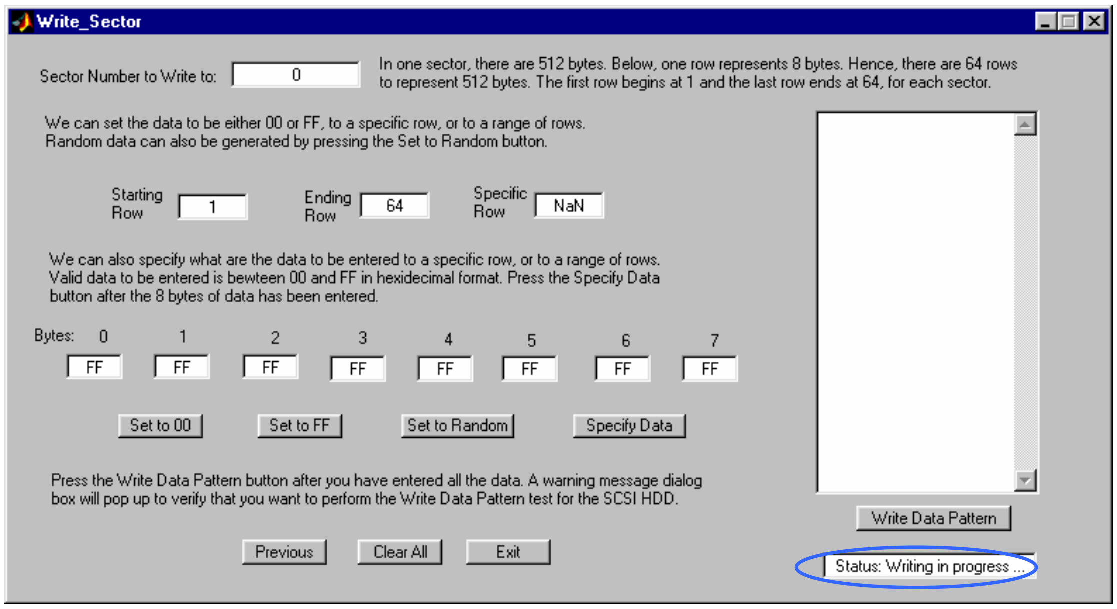

warning_msg

The Write Data Pattern test is DESTRUCTIVE.
It will erase all your data in the HDD.

Do you wish to proceed?

Yes | No

Below is an illustration of what will be displayed when hexadecimal data 'FF' is being written to a Sector Number 0.



Write_Sector

Sector Number to Write to: 0

In one sector, there are 512 bytes. Below, one row represents 8 bytes. Hence, there are 64 rows to represent 512 bytes. The first row begins at 1 and the last row ends at 64, for each sector.

We can set the data to be either 00 or FF, to a specific row, or to a range of rows. Random data can also be generated by pressing the Set to Random button.

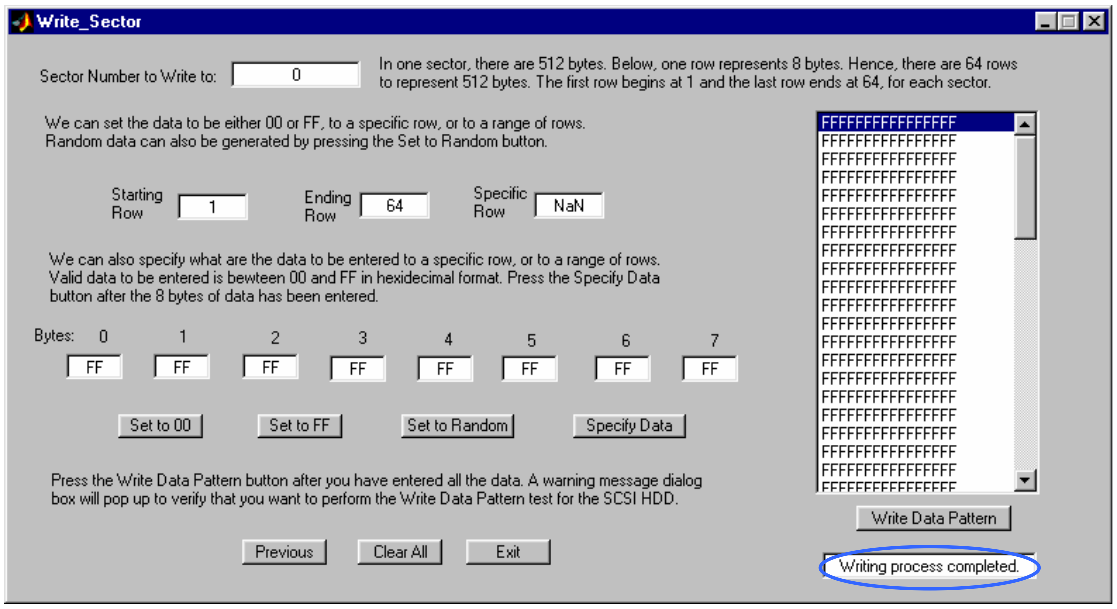
Starting Row: 1 Ending Row: 64 Specific Row: NaN

We can also specify what are the data to be entered to a specific row, or to a range of rows. Valid data to be entered is bewteen 00 and FF in hexidecimal format. Press the Specify Data button after the 8 bytes of data has been entered.

Bytes:	0	1	2	3	4	5	6	7
	FF	FF	FF	FF	FF	FF	FF	FF

Press the Write Data Pattern button after you have entered all the data. A warning message dialog box will pop up to verify that you want to perform the Write Data Pattern test for the SCSI HDD.

Status: Writing in progress ...



Write_Sector

Sector Number to Write to: 0

In one sector, there are 512 bytes. Below, one row represents 8 bytes. Hence, there are 64 rows to represent 512 bytes. The first row begins at 1 and the last row ends at 64, for each sector.

We can set the data to be either 00 or FF, to a specific row, or to a range of rows. Random data can also be generated by pressing the Set to Random button.

Starting Row: 1 Ending Row: 64 Specific Row: NaN

We can also specify what are the data to be entered to a specific row, or to a range of rows. Valid data to be entered is bewteen 00 and FF in hexidecimal format. Press the Specify Data button after the 8 bytes of data has been entered.

Bytes:	0	1	2	3	4	5	6	7
	FF	FF	FF	FF	FF	FF	FF	FF

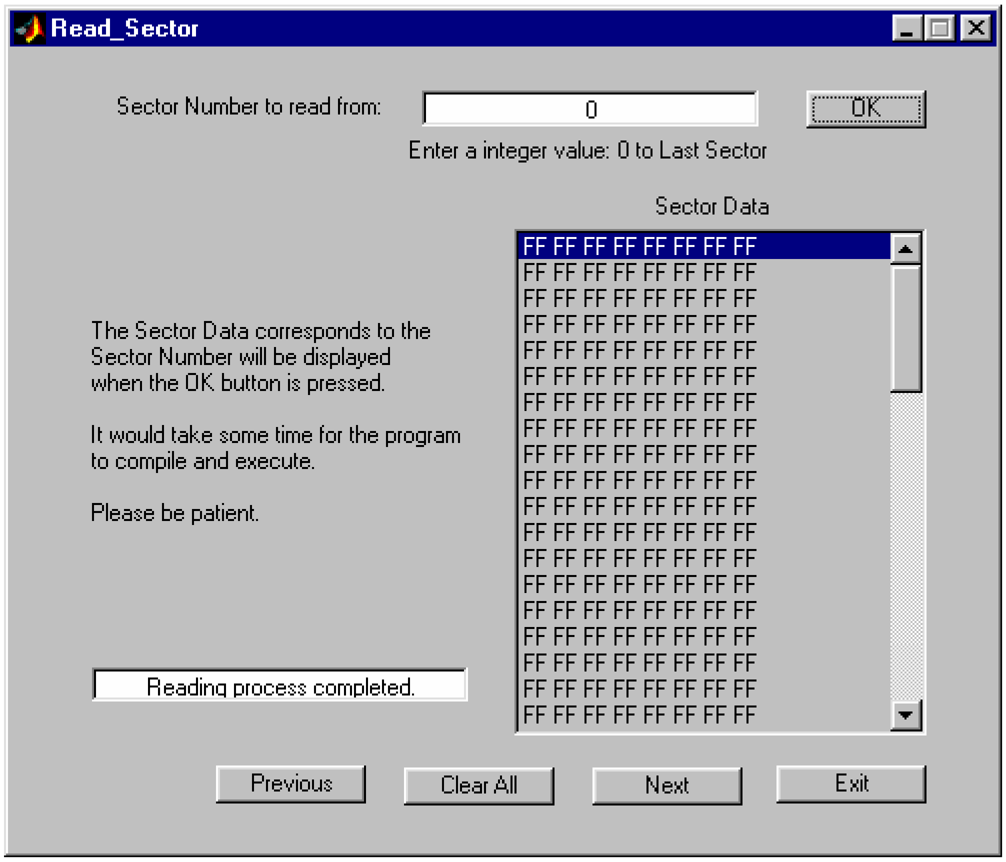
Press the Write Data Pattern button after you have entered all the data. A warning message dialog box will pop up to verify that you want to perform the Write Data Pattern test for the SCSI HDD.

```
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
```

Writing process completed.

To verify that the data being written for the particular sector is correct, click on 'Previous' pushbutton to return to the Read Sector dialog box, as shown in the next page.

The dialog box below verifies that the hexadecimal data 'FF' has been correctly written to the Sector Number 0.



Read_Sector

Sector Number to read from: 0

OK

Enter a integer value: 0 to Last Sector

Sector Data

```
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
```

The Sector Data corresponds to the Sector Number will be displayed when the OK button is pressed.

It would take some time for the program to compile and execute.

Please be patient.

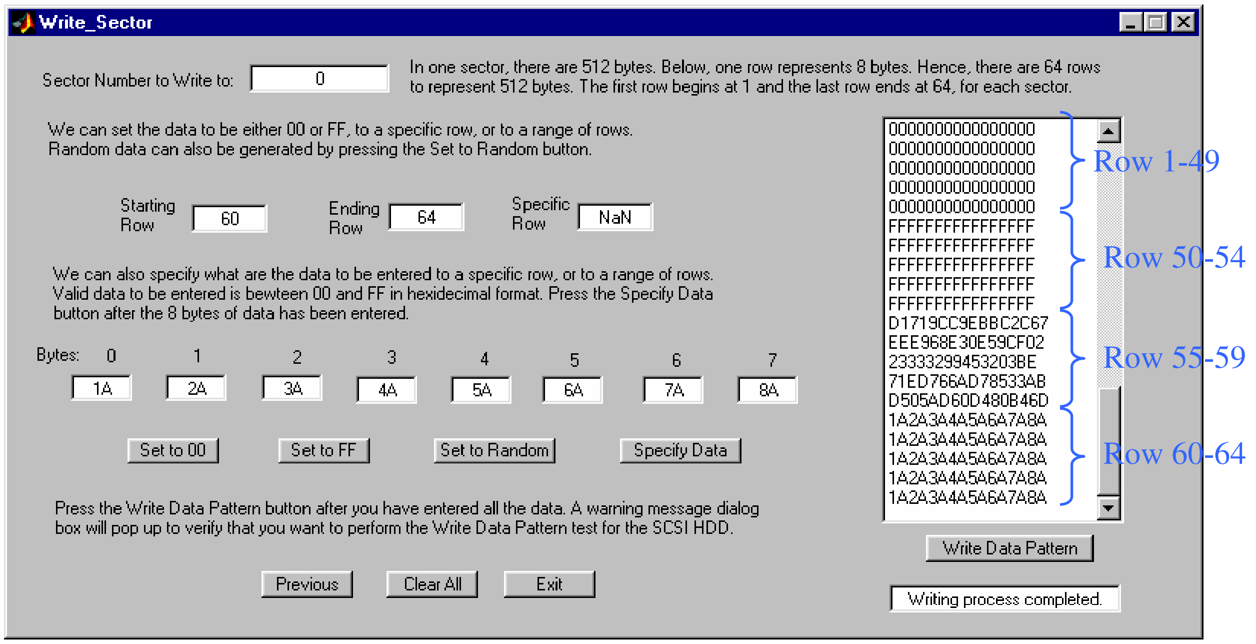
Reading process completed.

Above is a brief description of how to use the program to perform the Start, Stop, Read and Write functions for a SCSI HDD.

Next, we shall illustrate a particular example where the user can write data pattern of any kinds, say

(1) '00' data, to row 1 to row 49
(2) 'FF' data, to row 50 to row 54
(3) Random data, to row 55 to 59
(4) User-defined data, (1A, 2A, 3A, 4A, 5A, 6A, 7A, 8A), to row 60 to 64

into a particular Sector Number 0. (see next page)



Sector Number to Write to: 0

In one sector, there are 512 bytes. Below, one row represents 8 bytes. Hence, there are 64 rows to represent 512 bytes. The first row begins at 1 and the last row ends at 64, for each sector.

We can set the data to be either 00 or FF, to a specific row, or to a range of rows. Random data can also be generated by pressing the Set to Random button.

Starting Row: 60 Ending Row: 64 Specific Row: NaN

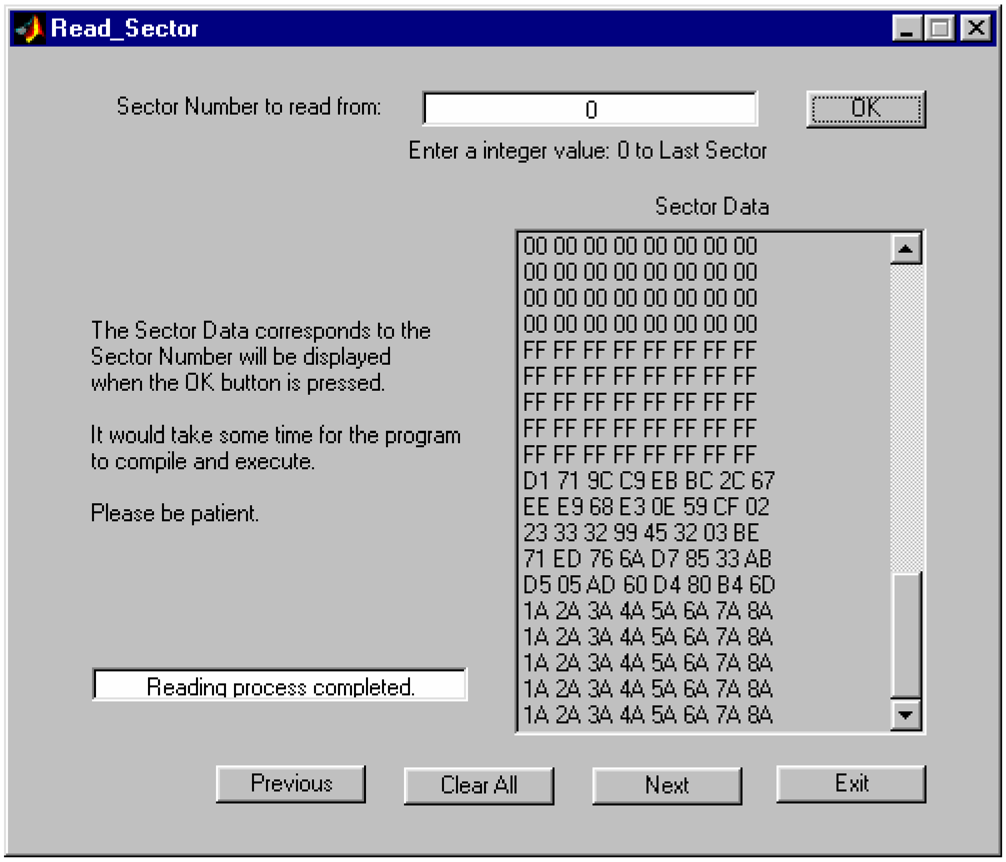
We can also specify what are the data to be entered to a specific row, or to a range of rows. Valid data to be entered is bewteen 00 and FF in hexidecimal format. Press the Specify Data button after the 8 bytes of data has been entered.

Bytes:	0	1	2	3	4	5	6	7
	1A	2A	3A	4A	5A	6A	7A	8A

Press the Write Data Pattern button after you have entered all the data. A warning message dialog box will pop up to verify that you want to perform the Write Data Pattern test for the SCSI HDD.

```
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
D1719CC9EBBC2C67
EEE968E30E59CF02
233332994532O3BE
71ED766AD78533AB
D505AD60D480B46D
1A2A3A4A5A6A7A8A
1A2A3A4A5A6A7A8A
1A2A3A4A5A6A7A8A
1A2A3A4A5A6A7A8A
1A2A3A4A5A6A7A8A
```

Writing process completed.



Sector Number to read from: 0

Enter a integer value: 0 to Last Sector

Sector Data

```
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
D1 71 9C C9 EB BC 2C 67
EE E9 68 E3 0E 59 CF 02
23 33 32 99 45 32 03 BE
71 ED 76 6A D7 85 33 AB
D5 05 AD 60 D4 80 B4 6D
1A 2A 3A 4A 5A 6A 7A 8A
1A 2A 3A 4A 5A 6A 7A 8A
1A 2A 3A 4A 5A 6A 7A 8A
1A 2A 3A 4A 5A 6A 7A 8A
1A 2A 3A 4A 5A 6A 7A 8A
```

The Sector Data corresponds to the Sector Number will be displayed when the OK button is pressed.

It would take some time for the program to compile and execute.

Please be patient.

Reading process completed.

The above two dialog boxes verify that the Sector Data has been written correctly into Sector Number 0.

Notes
Advanced SCSI Programming Interface (ASPI) for Win32 is used for the connection of most SCSI peripheral devices. It defines a protocol for the SCSI applications to submit I/O requests to a single operating system driver. Access to the operating system driver is made through the Dynamic Link Library, namely "wnaspi32.dll". Note here that "wnaspi32.dll" is written by Adaptec, not Microsoft.

The written software is tested both in Windows 98 and Windows 2000 systems. Make sure that the latest updated ASPI drivers version 4.71 for Windows 98, NT 4, Me, 2000 and XP are used. They can be downloaded at http://www.adaptec.com

Future Works
The implementation for the Read Sector and Write Sector functions is based on the 6-byte commands, namely (0x08) and (0x0A). Hence, there is a limitation on the maximum number of sectors which the above functions can access, that is, 2,097,151 (2^21-1) sectors of the HDD. Future work will explore on the 10-byte, 12-byte or perhaps 16-byte commands.

Also, as "wnaspi32.dll" maybe outdated soon, a better way of communicating with the SCSI devices is via the SCSI Pass-Through Interface (SPTI). The SCSI Pass-Through driver, which can be implemented for Windows NT and Windows 2000 as a class driver, is able to provide an IOCTL interface to the application to communicate with any SCSI device via DeviceIOControl function. We will look into the above for implementation soon.

Conclusion
The simple MATLAB program written is able to perform the functions of start, stop, read and write sector data for the Parallel SCSI HDD successfully.

Contacts
If you have any further questions or doubts in using the software, you are welcome to contact the following person:

(1) Mr Tan Thiam Huat, Email: th_scsi@yahoo.com

References
[1] The Book of SCSI, 2d Edition, by Gary Field, Peter Ridge, et al.

[2] The SCSI Bus & IDE Interface, Protocols, Applications & Programming, by Friedhelm Schmidt, Addison-Wesley, Second Edition, 1998.

[3] MATLAB Application Program Interface Reference, pdf

[4] Graphics and GUIs with MATLAB, Third Edition, by Patrick M. , O. Thomas Holland, 2002.

Disclaimer
As the write sector function is destructive and may result in data loss, the author will not take any responsibilities whatsoever.