The Linux SYM-2 driver documentation file

Written by Gerard Roudier <groudier@free.fr>
21 Rue Carnot
95170 DEUIL LA BARRE - FRANCE

Updated by Matthew Wilcox <matthew@wil.cx>

2004-10-09

==========

==========

1. Introduction

This driver supports the whole SYM53C8XX family of PCI-SCSI controllers.
It also support the subset of LSI53C10XX PCI-SCSI controllers that are based
on the SYM53C8XX SCRIPTS language.

It replaces the sym53c8xx+ncr53c8xx driver bundle and shares its core code
with the FreeBSD SYM-2 driver. The `glue' that allows this driver to work
under Linux is contained in 2 files named sym_glue.h and sym_glue.c.
Other drivers files are intended not to depend on the Operating System
on which the driver is used.

The history of this driver can be summarized as follows:

1993: ncr driver written for 386bsd and FreeBSD by:
        Wolfgang Stanglmeier   <wolf@cologne.de>
        Stefan Esser   <se@mi.Uni-Koeln.de>

1996: port of the ncr driver to Linux-1.2.13 and rename it ncr53c8xx.
        Gerard Roudier

1998: new sym53c8xx driver for Linux based on LOAD/STORE instruction and that
      adds full support for the 896 but drops support for early NCR devices.
        Gerard Roudier

1999: port of the sym53c8xx driver to FreeBSD and support for the LSI53C1010
      33 MHz and 66MHz Ultra-3 controllers. The new driver is named `sym'.
        Gerard Roudier

2000: Add support for early NCR devices to FreeBSD `sym' driver.
      Break the driver into several sources and separate the OS glue
      code from the core code that can be shared among different O/Ses.
      Write a glue code for Linux.
        Gerard Roudier

2004: Remove FreeBSD compatibility code. Remove support for versions of
      Linux before 2.6. Start using Linux facilities.

This README file addresses the Linux version of the driver. Under FreeBSD,
the driver documentation is the sym.8 man page.

Information about new chips is available at LSILOGIC web server:

        http://www.lsilogic.com/

SCSI standard documentations are available at T10 site:

        http://www.t10.org/

Useful SCSI tools written by Eric Youngdale are part of most Linux
distributions:
   scsiinfo: command line tool
   scsi-config: TCL/Tk tool using scsiinfo

2. Supported chips and SCSI features

The following features are supported for all chips:

- Synchronous negotiation
- Disconnection
- Tagged command queuing
- SCSI parity checking
- PCI Master parity checking

Other features depends on chip capabilities.
The driver notably uses optimized SCRIPTS for devices that support LOAD/STORE and handles PHASE MISMATCH from SCRIPTS for devices that support the corresponding feature.

The following table shows some characteristics of the chip family.

Chip	On board SDMS BIOS	Wide	SCSI std.	Max. sync	LOAD/STORE SCRIPTS	HARDWARE PHASE MISMATCH
810	N	N	FAST10	10 MB/s	N	N
810A	N	N	FAST10	10 MB/s	Y	N
815	Y	N	FAST10	10 MB/s	N	N
825	Y	Y	FAST10	20 MB/s	N	N
825A	Y	Y	FAST10	20 MB/s	Y	N
860	N	N	FAST20	20 MB/s	Y	N
875	Y	Y	FAST20	40 MB/s	Y	N
875A	Y	Y	FAST20	40 MB/s	Y	Y
876	Y	Y	FAST20	40 MB/s	Y	N
895	Y	Y	FAST40	80 MB/s	Y	N
895A	Y	Y	FAST40	80 MB/s	Y	Y
896	Y	Y	FAST40	80 MB/s	Y	Y
897	Y	Y	FAST40	80 MB/s	Y	Y
1510D	Y	Y	FAST40	80 MB/s	Y	Y
1010	Y	Y	FAST80	160 MB/s	Y	Y
1010_66*	Y	Y	FAST80	160 MB/s	Y	Y

* Chip supports 33MHz and 66MHz PCI bus clock.

Summary of other supported features:

```
Module:                 allow to load the driver
Memory mapped I/O:      increases performance
Control commands:       write operations to the proc SCSI file system
Debugging information: written to syslog (expert only)
Scatter / gather
Shared interrupt
Boot setup commands
Serial NVRAM:           Symbios and Tekram formats
```

3. Advantages of this driver for newer chips.

3.1 Optimized SCSI SCRIPTS.

All chips except the 810, 815 and 825, support new SCSI SCRIPTS instructions

named LOAD and STORE that allow to move up to 1 DWORD from/to an IO register to/from memory much faster that the MOVE MEMORY instruction that is supported by the 53c7xx and 53c8xx family.

The LOAD/STORE instructions support absolute and DSA relative addressing modes. The SCSI SCRIPTS had been entirely rewritten using LOAD/STORE instead of MOVE MEMORY instructions.

Due to the lack of LOAD/STORE SCRIPTS instructions by earlier chips, this driver also incorporates a different SCRIPTS set based on MEMORY MOVE, in order to provide support for the entire SYM53C8XX chips family.

3.2 New features appeared with the SYM53C896

Newer chips (see above) allows handling of the phase mismatch context from SCRIPTS (avoids the phase mismatch interrupt that stops the SCSI processor until the C code has saved the context of the transfer).

The 896 and 1010 chips support 64 bit PCI transactions and addressing, while the 895A supports 32 bit PCI transactions and 64 bit addressing. The SCRIPTS processor of these chips is not true 64 bit, but uses segment registers for bit 32-63. Another interesting feature is that LOAD/STORE instructions that address the on-chip RAM (8k) remain internal to the chip.

4. Memory mapped I/O versus normal I/O

Memory mapped I/O has less latency than normal I/O and is the recommended way for doing IO with PCI devices. Memory mapped I/O seems to work fine on most hardware configurations, but some poorly designed chipsets may break this feature. A configuration option is provided for normal I/O to be used but the driver defaults to MMIO.

5. Tagged command queueing

Queuing more than 1 command at a time to a device allows it to perform optimizations based on actual head positions and its mechanical characteristics. This feature may also reduce average command latency. In order to really gain advantage of this feature, devices must have a reasonable cache size (No miracle is to be expected for a low-end hard disk with 128 KB or less).
Some known old SCSI devices do not properly support tagged command queuing. Generally, firmware revisions that fix this kind of problems are available at respective vendor web/ftp sites.
All I can say is that I never have had problem with tagged queuing using this driver and its predecessors. Hard disks that behaved correctly for me using tagged commands are the following:

- IBM S12 0662
- Conner 1080S
- Quantum Atlas I
- Quantum Atlas II
- Seagate Cheetah I
- Quantum Viking II
- IBM DRVS
- Quantum Atlas IV
- Seagate Cheetah II

If your controller has NVRAM, you can configure this feature per target from the user setup tool. The Tekram Setup program allows to tune the maximum number of queued commands up to 32. The Symbios Setup only allows to enable or disable this feature.

The maximum number of simultaneous tagged commands queued to a device is currently set to 16 by default. This value is suitable for most SCSI disks. With large SCSI disks (>= 2GB, cache >= 512KB, average seek time <= 10 ms), using a larger value may give better performances.

This driver supports up to 255 commands per device, and but using more than 64 is generally not worth-while, unless you are using a very large disk or disk arrays. It is noticeable that most of recent hard disks seem not to accept more than 64 simultaneous commands. So, using more than 64 queued commands is probably just resource wasting.

If your controller does not have NVRAM or if it is managed by the SDMS BIOS/SETUP, you can configure tagged queueing feature and device queue depths from the boot command-line. For example:

```
  sym53c8xx=tags:4/t2t3q15-t4q7/t1u0q32
```

will set tagged commands queue depths as follow:

```
- target 2  all luns  on controller 0 --> 15
- target 3  all luns  on controller 0 --> 15
- target 4  all luns  on controller 0 -->  7
- target 1  lun 0     on controller 1 --> 32
- all other target/lun                -->  4
```

In some special conditions, some SCSI disk firmwares may return a QUEUE FULL status for a SCSI command. This behaviour is managed by the driver using the following heuristic:

- Each time a QUEUE FULL status is returned, tagged queue depth is reduced to the actual number of disconnected commands.

- Every 200 successfully completed SCSI commands, if allowed by the current limit, the maximum number of queueable commands is incremented.

Since QUEUE FULL status reception and handling is resource wasting, the driver notifies by default this problem to user by indicating the actual number of commands used and their status, as well as its decision on the device queue depth change.
The heuristic used by the driver in handling QUEUE FULL ensures that the impact on performances is not too bad. You can get rid of the messages by setting verbose level to zero, as follow:

```
1st method: boot your system using 'sym53c8xx=verb:0' option.
2nd method: apply "setverbose 0" control command to the proc fs entry
            corresponding to your controller after boot-up.
```

6. Parity checking

The driver supports SCSI parity checking and PCI bus master parity

checking. These features must be enabled in order to ensure safe
data transfers. Some flawed devices or mother boards may have problems
with parity. The options to defeat parity checking have been removed
from the driver.

7. Profiling information

This driver does not provide profiling informations as did its predecessors.
This feature was not this useful and added complexity to the code.
As the driver code got more complex, I have decided to remove everything
that didn't seem actually useful.

8. Control commands

Control commands can be sent to the driver with write operations to
the proc SCSI file system. The generic command syntax is the
following:

```
echo "<verb> <parameters>" >/proc/scsi/sym53c8xx/0
(assumes controller number is 0)
```

Using "all" for "<target>" parameter with the commands below will
apply to all targets of the SCSI chain (except the controller).

Available commands:

8.1 Set minimum synchronous period factor

```
setsync <target> <period factor>

target:    target number
period:    minimum synchronous period.
           Maximum speed = 1000/(4*period factor) except for special
           cases below.

Specify a period of 0, to force asynchronous transfer mode.

  9 means 12.5 nano-seconds synchronous period
 10 means 25 nano-seconds synchronous period
 11 means 30 nano-seconds synchronous period
 12 means 50 nano-seconds synchronous period
```

8.2 Set wide size

```
setwide <target> <size>

target:    target number
size:      0=8 bits, 1=16bits
```

8.3 Set maximum number of concurrent tagged commands

```
settags <target> <tags>

target:    target number
tags:      number of concurrent tagged commands
           must not be greater than configured (default: 16)
```

8.4 Set debug mode

```
setdebug <list of debug flags>

Available debug flags:
    alloc:   print info about memory allocations (ccb, lcb)
    queue:   print info about insertions into the command start queue
    result:  print sense data on CHECK CONDITION status
    scatter: print info about the scatter process
    scripts: print info about the script binding process
    tiny:    print minimal debugging information
    timing:  print timing information of the NCR chip
    nego:    print information about SCSI negotiations
    phase:   print information on script interruptions

Use "setdebug" with no argument to reset debug flags.
```

8.5 Set flag (no_disc)

```
setflag <target> <flag>

target:    target number

For the moment, only one flag is available:

    no_disc:   not allow target to disconnect.

Do not specify any flag in order to reset the flag. For example:
- setflag 4
  will reset no_disc flag for target 4, so will allow it disconnections.
- setflag all
  will allow disconnection for all devices on the SCSI bus.
```

8.6 Set verbose level

```
setverbose #level

The driver default verbose level is 1. This command allows to change
th driver verbose level after boot-up.
```

8.7 Reset all logical units of a target

```
resetdev <target>

target:    target number
The driver will try to send a BUS DEVICE RESET message to the target.
```

8.8 Abort all tasks of all logical units of a target

```
cleardev <target>

target:    target number
The driver will try to send a ABORT message to all the logical units
```

of the target.

9. Configuration parameters

Under kernel configuration tools (make menuconfig, for example), it is possible to change some default driver configuration parameters. If the firmware of all your devices is perfect enough, all the features supported by the driver can be enabled at start-up. However, if only one has a flaw for some SCSI feature, you can disable the support by the driver of this feature at linux start-up and enable this feature after boot-up only for devices that support it safely.

Configuration parameters:

Use normal IO (default answer: n)
Answer "y" if you suspect your mother board to not allow memory mapped I/O.
May slow down performance a little.

Default tagged command queue depth (default answer: 16)
Entering 0 defaults to tagged commands not being used.
This parameter can be specified from the boot command line.

Maximum number of queued commands (default answer: 32)
This option allows you to specify the maximum number of tagged commands that can be queued to a device. The maximum supported value is 255.

Synchronous transfers frequency (default answer: 80)
This option allows you to specify the frequency in MHz the driver will use at boot time for synchronous data transfer negotiations. 0 means "asynchronous data transfers".

10. Boot setup commands

10.1 Syntax

Setup commands can be passed to the driver either at boot time or as parameters to modprobe, as described in Documentation/kernel-parameters.txt

Example of boot setup command under lilo prompt:

lilo: linux root=/dev/sda2 sym53c8xx.cmd_per_lun=4 sym53c8xx.sync=10 sym53c8xx.debug=0x200

- enable tagged commands, up to 4 tagged commands queued.
- set synchronous negotiation speed to 10 Mega-transfers / second.
- set DEBUG_NEGO flag.

The following command will install the driver module with the same options as above.

modprobe sym53c8xx cmd_per_lun=4 sync=10 debug=0x200

10.2 Available arguments

10.2.1 Default number of tagged commands

```
        cmd_per_lun=0 (or cmd_per_lun=1) tagged command queuing disabled
        cmd_per_lun=#tags (#tags > 1) tagged command queuing enabled
  #tags will be truncated to the max queued commands configuration parameter.

10.2.2 Burst max
        burst=0    burst disabled
        burst=255  get burst length from initial IO register settings.
        burst=#x   burst enabled (1<<#x burst transfers max)
  #x is an integer value which is log base 2 of the burst transfers max.
  By default the driver uses the maximum value supported by the chip.

10.2.3 LED support
        led=1      enable  LED support
        led=0      disable LED support
  Do not enable LED support if your scsi board does not use SDMS BIOS.
  (See 'Configuration parameters')

10.2.4 Differential mode
        diff=0  never set up diff mode
        diff=1  set up diff mode if BIOS set it
        diff=2  always set up diff mode
        diff=3  set diff mode if GPIO3 is not set

10.2.5 IRQ mode
        irqm=0     always open drain
        irqm=1     same as initial settings (assumed BIOS settings)
        irqm=2     always totem pole

10.2.6 Check SCSI BUS
        buschk=<option bits>

    Available option bits:
        0x0:   No check.
        0x1:   Check and do not attach the controller on error.
        0x2:   Check and just warn on error.

10.2.7 Suggest a default SCSI id for hosts
        hostid=255      no id suggested.
        hostid=#x   (0 < x < 7) x suggested for hosts SCSI id.

    If a host SCSI id is available from the NVRAM, the driver will ignore
    any value suggested as boot option. Otherwise, if a suggested value
    different from 255 has been supplied, it will use it. Otherwise, it will
    try to deduce the value previously set in the hardware and use value
    7 if the hardware value is zero.

10.2.8  Verbosity level
        verb=0     minimal
        verb=1     normal
        verb=2     too much

10.2.9 Debug mode
        debug=0  clear debug flags
        debug=#x   set debug flags
  #x is an integer value combining the following power-of-2 values:
  DEBUG_ALLOC       0x1
```

```
DEBUG_PHASE       0x2
DEBUG_POLL        0x4
DEBUG_QUEUE       0x8
DEBUG_RESULT     0x10
DEBUG_SCATTER    0x20
DEBUG_SCRIPT     0x40
DEBUG_TINY       0x80
DEBUG_TIMING    0x100
DEBUG_NEGO      0x200
DEBUG_TAGS      0x400
DEBUG_FREEZE    0x800
DEBUG_RESTART  0x1000
```

You can play safely with DEBUG_NEGO. However, some of these flags may generate bunches of syslog messages.

10.2.10 Settle delay

```
settle=n        delay for n seconds
```

After a bus reset, the driver will delay for n seconds before talking to any device on the bus. The default is 3 seconds and safe mode will default it to 10.

10.2.11 Serial NVRAM

```
NB: option not currently implemented.
nvram=n     do not look for serial NVRAM
nvram=y     test controllers for onboard serial NVRAM
(alternate binary form)
nvram=<bits options>
0x01   look for NVRAM  (equivalent to nvram=y)
0x02   ignore NVRAM "Synchronous negotiation" parameters for all devices
0x04   ignore NVRAM "Wide negotiation"  parameter for all devices
0x08   ignore NVRAM "Scan at boot time" parameter for all devices
0x80   also attach controllers set to OFF in the NVRAM (sym53c8xx only)
```

10.2.12 Exclude a host from being attached

```
excl=<io_address>,...
```

Prevent host at a given io address from being attached. For example 'excl=0xb400,0xc000' indicate to the driver not to attach hosts at address 0xb400 and 0xc000.

10.3 Converting from old style options

Previously, the sym2 driver accepted arguments of the form

```
sym53c8xx=tags:4,sync:10,debug:0x200
```

As a result of the new module parameters, this is no longer available. Most of the options have remained the same, but tags has become cmd_per_lun to reflect its different purposes. The sample above would be specified as:

```
modprobe sym53c8xx cmd_per_lun=4 sync=10 debug=0x200
```

or on the kernel boot line as:

```
sym53c8xx.cmd_per_lun=4 sym53c8xx.sync=10 sym53c8xx.debug=0x200
```

10.4 SCSI BUS checking boot option.

When this option is set to a non-zero value, the driver checks SCSI lines logic state, 100 micro-seconds after having asserted the SCSI RESET line. The driver just reads SCSI lines and checks all lines read FALSE except RESET. Since SCSI devices shall release the BUS at most 800 nano-seconds after SCSI RESET has been asserted, any signal to TRUE may indicate a SCSI BUS problem. Unfortunately, the following common SCSI BUS problems are not detected:
- Only 1 terminator installed.
- Misplaced terminators.
- Bad quality terminators.
On the other hand, either bad cabling, broken devices, not conformant devices, ... may cause a SCSI signal to be wrong when te driver reads it.

15. SCSI problem troubleshooting

15.1 Problem tracking

Most SCSI problems are due to a non conformant SCSI bus or too buggy devices. If unfortunately you have SCSI problems, you can check the following things:

- SCSI bus cables
- terminations at both end of the SCSI chain
- linux syslog messages (some of them may help you)

If you do not find the source of problems, you can configure the driver or devices in the NVRAM with minimal features.

- only asynchronous data transfers
- tagged commands disabled
- disconnections not allowed

Now, if your SCSI bus is ok, your system has every chance to work with this safe configuration but performances will not be optimal.

If it still fails, then you can send your problem description to appropriate mailing lists or news-groups. Send me a copy in order to be sure I will receive it. Obviously, a bug in the driver code is possible.

My current email address: Gerard Roudier <groudier@free.fr>

Allowing disconnections is important if you use several devices on your SCSI bus but often causes problems with buggy devices. Synchronous data transfers increases throughput of fast devices like hard disks. Good SCSI hard disks with a large cache gain advantage of tagged commands queuing.

15.2 Understanding hardware error reports

When the driver detects an unexpected error condition, it may display a message of the following pattern.

sym0:1: ERROR (0:48) (1-21-65) (f/95/0) @ (script 7c0:19000000).
sym0: script cmd = 19000000

```
sym0: regdump: da 10 80 95 47 0f 01 07 75 01 81 21 80 01 09 00.
```

Some fields in such a message may help you understand the cause of the problem, as follows:

```
sym0:1: ERROR (0:48) (1-21-65) (f/95/0) @ (script 7c0:19000000).
.....A.........B.C....D.E..F....G.H..I.......J.....K...L.......
```

Field A : target number.
  SCSI ID of the device the controller was talking with at the moment the error occurs.

Field B : DSTAT io register (DMA STATUS)
  Bit 0x40 : MDPE Master Data Parity Error
             Data parity error detected on the PCI BUS.
  Bit 0x20 : BF   Bus Fault
             PCI bus fault condition detected
  Bit 0x01 : IID Illegal Instruction Detected
             Set by the chip when it detects an Illegal Instruction format
             on some condition that makes an instruction illegal.
  Bit 0x80 : DFE Dma Fifo Empty
             Pure status bit that does not indicate an error.
  If the reported DSTAT value contains a combination of MDPE (0x40), BF (0x20), then the cause may be likely due to a PCI BUS problem.

Field C : SIST io register (SCSI Interrupt Status)
  Bit 0x08 : SGE SCSI GROSS ERROR
             Indicates that the chip detected a severe error condition
             on the SCSI BUS that prevents the SCSI protocol from functioning
             properly.
  Bit 0x04 : UDC Unexpected Disconnection
             Indicates that the device released the SCSI BUS when the chip
             was not expecting this to happen. A device may behave so to
             indicate the SCSI initiator that an error condition not reportable
            using the SCSI protocol has occurred.
  Bit 0x02 : RST SCSI BUS Reset
             Generally SCSI targets do not reset the SCSI BUS, although any
             device on the BUS can reset it at any time.
  Bit 0x01 : PAR Parity
             SCSI parity error detected.
  On a faulty SCSI BUS, any error condition among SGE (0x08), UDC (0x04) and PAR (0x01) may be detected by the chip. If your SCSI system sometimes encounters such error conditions, especially SCSI GROSS ERROR, then a SCSI BUS problem is likely the cause of these errors.

For fields D,E,F,G and H, you may look into the sym53c8xx_defs.h file that contains some minimal comments on IO register bits.
Field D : SOCL Scsi Output Control Latch
          This register reflects the state of the SCSI control lines the
          chip want to drive or compare against.
Field E : SBCL Scsi Bus Control Lines
          Actual value of control lines on the SCSI BUS.
Field F : SBDL Scsi Bus Data Lines
          Actual value of data lines on the SCSI BUS.
Field G : SXFER SCSI Transfer
          Contains the setting of the Synchronous Period for output and

the current Synchronous offset (offset 0 means asynchronous).
Field H : SCNTL3 Scsi Control Register 3
Contains the setting of timing values for both asynchronous and synchronous data transfers.
Field I : SCNTL4 Scsi Control Register 4
Only meaningful for 53C1010 Ultra3 controllers.

Understanding Fields J, K, L and dumps requires to have good knowledge of SCSI standards, chip cores functionnals and internal driver data structures. You are not required to decode and understand them, unless you want to help maintain the driver code.

17. Serial NVRAM (added by Richard Waltham: dormouse@farsrobt.demon.co.uk)

17.1 Features

Enabling serial NVRAM support enables detection of the serial NVRAM included on Symbios and some Symbios compatible host adaptors, and Tekram boards. The serial NVRAM is used by Symbios and Tekram to hold set up parameters for the host adaptor and its attached drives.

The Symbios NVRAM also holds data on the boot order of host adaptors in a system with more than one host adaptor. This information is no longer used as it's fundamentally incompatible with the hotplug PCI model.

Tekram boards using Symbios chips, DC390W/F/U, which have NVRAM are detected and this is used to distinguish between Symbios compatible and Tekram host adaptors. This is used to disable the Symbios compatible "diff" setting incorrectly set on Tekram boards if the CONFIG_SCSI_53C8XX_SYMBIOS_COMPAT configuration parameter is set enabling both Symbios and Tekram boards to be used together with the Symbios cards using all their features, including "diff" support. ("led pin" support for Symbios compatible cards can remain enabled when using Tekram cards. It does nothing useful for Tekram host adaptors but does not cause problems either.)

The parameters the driver is able to get from the NVRAM depend on the data format used, as follow:

	Tekram format	Symbios format
General and host parameters		
Boot order	N	Y
Host SCSI ID	Y	Y
SCSI parity checking	Y	Y
Verbose boot messages	N	Y
SCSI devices parameters		
Synchronous transfer speed	Y	Y
Wide 16 / Narrow	Y	Y
Tagged Command Queuing enabled	Y	Y
Disconnections enabled	Y	Y
Scan at boot time	N	Y

In order to speed up the system boot, for each device configured without the "scan at boot time" option, the driver forces an error on the first TEST UNIT READY command received for this device.

17.2 Symbios NVRAM layout

typical data at NVRAM address 0x100 (53c810a NVRAM)

```
---------------------------------------------------------
00 00
64 01
8e 0b

00 30 00 00 00 00 07 00 00 00 00 00 00 00 07 04 10 04 00 00

04 00 0f 00 00 10 00 50 00 00 01 00 00 62
04 00 03 00 00 10 00 58 00 00 01 00 00 63
04 00 01 00 00 10 00 48 00 00 01 00 00 61
00 00 00 00 00 00 00 00 00 00 00 00 00 00

0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00

0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 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 00

fe fe
```

```
00 00
00 00
```

NVRAM layout details

```
NVRAM Address 0x000-0x0ff not used
              0x100-0x26f initialised data
              0x270-0x7ff not used
```

general layout

```
        header  -   6 bytes,
        data    - 356 bytes (checksum is byte sum of this data)
        trailer -   6 bytes
                  ---
        total     368 bytes
```

data area layout

```
        controller set up  -  20 bytes
        boot configuration -  56 bytes (4x14 bytes)
        device set up      - 128 bytes (16x8 bytes)
        unused (spare?)    - 152 bytes (19x8 bytes)
                             ---
        total                356 bytes
```

---

header

```
00 00   - ?? start marker
64 01   - byte count (lsb/msb excludes header/trailer)
8e 0b   - checksum (lsb/msb excludes header/trailer)
```

---

controller set up

```
00 30 00 00 00 00 07 00 00 00 00 00 00 00 07 04 10 04 00 00
                     |     |           |     |
                     |     |           |     -- host ID
                     |     |           |
                     |     |           --Removable Media Support
                     |     |              0x00 = none
                     |     |              0x01 = Bootable Device
                     |     |              0x02 = All with Media
                     |     |
                     |     --flag bits 2
                     |       0x00000001= scan order hi->low
                     |           (default 0x00 - scan low->hi)
                     --flag bits 1
                        0x00000001 scam enable
                        0x00000010 parity enable
                        0x00000100 verbose boot msgs
```

remaining bytes unknown - they do not appear to change in my
current set up for any of the controllers.

default set up is identical for 53c810a and 53c875 NVRAM

(Removable Media added Symbios BIOS version 4.09)

---

boot configuration

boot order set by order of the devices in this table

```
04 00 0f 00 00 10 00 50 00 00 01 00 00 62 -- 1st controller
04 00 03 00 00 10 00 58 00 00 01 00 00 63    2nd controller
04 00 01 00 00 10 00 48 00 00 01 00 00 61    3rd controller
00 00 00 00 00 00 00 00 00 00 00 00 00 00    4th controller
        |  |  |  |  |  |  |  |  |  |  |  |
        |  |  |  |  |  |  |  |  |  |  |   ---- PCI io port adr
        |  |  |  |  |  |  |  |  |  |   --0x01 init/scan at boot time
        |  |  |  |  |  |  |  |  |   --PCI device/function number (0xdddddfff)
        |  |  |  |  |  |  |  |   ----- ?? PCI vendor ID (lsb/msb)
        |  |  |  |  |  |  |   ----PCI device ID (lsb/msb)
```

?? use of this data is a guess but seems reasonable

remaining bytes unknown - they do not appear to change in my current set up

default set up is identical for 53c810a and 53c875 NVRAM

---

device set up (up to 16 devices - includes controller)

```
0f 00 08 08 64 00 0a 00 - id 0
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00

0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00
0f 00 08 08 64 00 0a 00 - id 15
  |  |  |  |  |  |  |  |
  |  |  |  |  |  |  |   ----timeout (lsb/msb)
  |  |  |   --synch period (0x?? 40 Mtrans/sec- fast 40) (probably 0x28)
  |  |  |                  (0x30 20 Mtrans/sec- fast 20)
  |  |  |                  (0x64 10 Mtrans/sec- fast )
  |  |  |                  (0xc8  5 Mtrans/sec)
  |  |  |                  (0x00  asynchronous)
  |  |   -- ?? max sync offset (0x08 in NVRAM on 53c810a)
  |  |                         (0x10 in NVRAM on 53c875)
  |  |  --device bus width (0x08 narrow)
  |  |                     (0x10 16 bit wide)
  |  |
  |   --flag bits
  |     0x00000001 - disconnect enabled
```

```
    0x00000010 - scan at boot time
    0x00000100 - scan luns
    0x00001000 - queue tags enabled

remaining bytes unknown - they do not appear to change in my
current set up

?? use of this data is a guess but seems reasonable
(but it could be max bus width)

default set up for 53c810a NVRAM
default set up for 53c875 NVRAM - bus width     - 0x10
                                - sync offset ? - 0x10
                                - sync period   - 0x30
------------------------------------------------------------
?? spare device space (32 bit bus ??)

00 00 00 00 00 00 00 00  (19x8bytes)
.
.
00 00 00 00 00 00 00 00

default set up is identical for 53c810a and 53c875 NVRAM
------------------------------------------------------------
trailer

fe fe   - ? end marker ?
00 00
00 00

default set up is identical for 53c810a and 53c875 NVRAM
------------------------------------------------------------



17.3 Tekram NVRAM layout

nvram 64x16 (1024 bit)

Drive settings

Drive ID 0-15 (addr 0x0yyyy0 = device setup, yyyy = ID)
              (addr 0x0yyyy1 = 0x0000)

    x x x x  x x x x  x x x x  x x x x
                                  |
                                   ----- parity check   0 - off
                                                        1 - on
                                 |
                                  ------- sync neg      0 - off
                                                        1 - on
                                |
                                 --------- disconnect   0 - off
                                                        1 - on
                               |
                                ----------- start cmd    0 - off
```

```
               | | |      | |                              1 - on
               | | |      | |
               | | |      |  -------------- tagged cmds    0 - off
               | | |      |                                1 - on
               | | |      |
               | | |       ---------------- wide neg       0 - off
               | | |                                       1 - on
               | | |
                --------------------------- sync rate      0 - 10.0 Mtrans/sec
                                                           1 -  8.0
                                                           2 -  6.6
                                                           3 -  5.7
                                                           4 -  5.0
                                                           5 -  4.0
                                                           6 -  3.0
                                                           7 -  2.0
                                                           7 -  2.0
                                                           8 - 20.0
                                                           9 - 16.7
                                                           a - 13.9
                                                           b - 11.9
```

Global settings

Host flags 0 (addr 0x100000, 32)

```
    x x x x  x x x x  x x x x  x x x x
    | | | |  | | | |           | | | |
    | | | |  | | | |            ----------- host ID    0x00 - 0x0f
    | | | |  | | | |
    | | | |  | | |  ----------------------- support for    0 - off
    | | | |  | | |                          > 2 drives     1 - on
    | | | |  | | |
    | | | |  | |  ------------------------- support drives 0 - off
    | | | |  | |                            > 1Gbytes      1 - on
    | | | |  | |
    | | | |  |  --------------------------- bus reset on   0 - off
    | | | |  |                                power on     1 - on
    | | | |  |
    | | | |   ----------------------------- active neg     0 - off
    | | | |                                                1 - on
    | | | |
    | | |  -------------------------------- imm seek       0 - off
    | | |                                                  1 - on
    | | |
    | |  ---------------------------------- scan luns      0 - off
    | |                                                    1 - on
    | |
     -------------------------------------- removable      0 - disable
                                            as BIOS dev    1 - boot device
                                                           2 - all
```

Host flags 1 (addr 0x100001, 33)

```
    x x x x  x x x x  x x x x  x x x x
               | | |             | | |
```

```
|  |  |              --------- boot delay     0 -   3 sec
|  |  |                                       1 -   5
|  |  |                                       2 -  10
|  |  |                                       3 -  20
|  |  |                                       4 -  30
|  |  |                                       5 -  60
|  |  |                                       6 - 120
|  |  |
 ---------------------------- max tag cmds    0 -  2
                                              1 -  4
                                              2 -  8
                                              3 - 16
                                              4 - 32
```

Host flags 2 (addr 0x100010, 34)

```
    x x x x  x x x x  x x x x  x x x x
                                  |
                                   ----- F2/F6 enable   0 - off ???
                                                        1 - on  ???
```

checksum (addr 0x111111)

checksum = 0x1234 - (sum addr 0-63)

---

default nvram data:

```
0x0037 0x0000 0x0037 0x0000 0x0037 0x0000 0x0037 0x0000
0x0037 0x0000 0x0037 0x0000 0x0037 0x0000 0x0037 0x0000
0x0037 0x0000 0x0037 0x0000 0x0037 0x0000 0x0037 0x0000
0x0037 0x0000 0x0037 0x0000 0x0037 0x0000 0x0037 0x0000

0x0f07 0x0400 0x0001 0x0000 0x0000 0x0000 0x0000 0x0000
0x0000 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x0000 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x0000 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000 0xfbbc
```

==========

End of Linux SYM-2 driver documentation file