USE OF EXAR'S ST16C2550 WITH LINUX 2.4.X & 2.6.X OPERATING SYSTEMS, PHOENIX BIOS VERSION 4.0 RELEASE 6.0 AND WINDOWS OPERATING SYSTEMS

1.0 INTRODUCTION

The following information is an application consideration in regards to the use of the ST16C2550 family of devices (ST16C2450, ST16C2550 and ST16C2552) with top mark date code of "A2 YYWW". This family will be referred to as the ST16C2550 "A2" device in the remainder of this application note.

2.0 PROBLEM DESCRIPTION

The ST16C2550 version "A2" shares a mask set with another UART device. Inadvertantly, the mask utilized included the Device ID (DVID), Device Revision (DREV) and Enhanced Feature (EFR) registers that should have been absent. These registers were not part of the previous ST16C2550 device with date code of "CC YY-WW".

The Linux Operating System (O/S) and the Phoenix BIOS issues have been attributed to the presence of these registers in the "A2" version of the device.

The "A2" device has been verified to operate correctly with the following operating systems: Windows 98, NT4.0, CE, 2000 and XP.

For those customers who are not impacted by this notice, please note that Exar plans on replacing the "A2" device with a future "B2" version.

Please see Table 1 for a summary of the affected parts on page 9 and "Q&A" on page 10.

DEVICE ID (DVID) AND DEVICE REVISION (DREV) REGISTERS

In the "A2" device version, DLM and DLL registers become the DVID and DREV registers when a value of 0x00 is written to both registers:

- The DLL register becomes the DREV register and shows the current device revision of 0x01.
- The DLM register becomes the DVID register and shows the device identification of 0x02.

ENHANCED FEATURE REGISTER (EFR)

The EFR register is only available in Exar's enhanced UART products such as the ST16C65x, XR16L255x, XR16L275x, XR16C85x, XR16C285x and XR16L78x. This register can be accessed by writing a value of 0xBF to the Line Control Register (LCR).

2.1 LINUX KERNEL 2.4.X AND 2.6.X DATA ERRORS

The Linux kernel (2.4.x and 2.6.x) identifies the ST16C2550 "A2" as a 16C650A due to the presence of the EFR register and the UART type detection algorithm. As shown in Figure 1A, the kernel defaults to the 16C650A (or 16C650V2) UART type when:

- it detects the presence of the EFR register, and
- it does not detect a 16C850 UART type or a 64-byte FIFO.

Operating the device as a 16C650V2 will cause an overrun condition resulting in data errors, due to the absence of 32-byte FIFO. This issue can be resolved if the algorithm is modified as shown in Figure 1B.

FIGURE 1. ORIGINAL AND MODIFIED UART TYPE DETECTION IN LINUX OPERATING SYSTEM KERNELS 2.4.X & 2.6.X

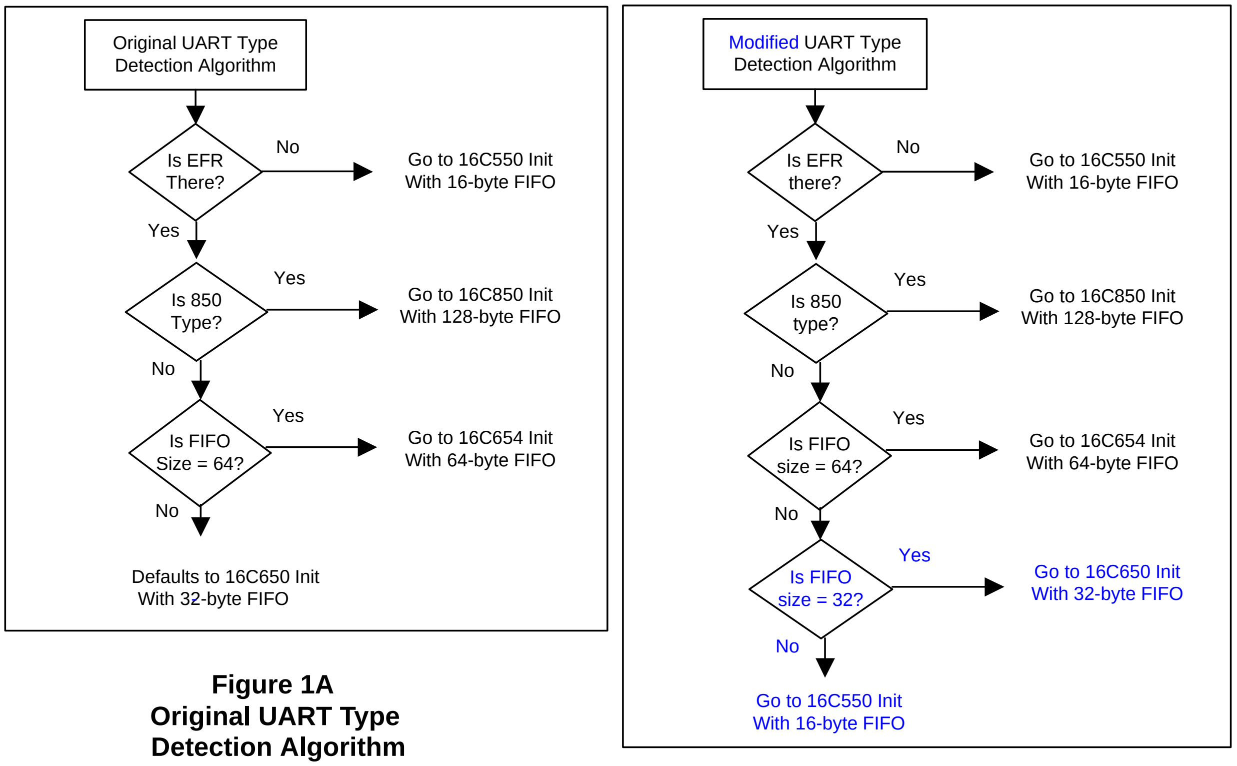
Original UART Type Detection Algorithm
Is EFR There?
No
Go to 16C550 Init With 16-byte FIFO
Yes
Is 850 Type?
Yes
Go to 16C850 Init With 128-byte FIFO
No
Is FIFO Size = 64?
Yes
Go to 16C654 Init With 64-byte FIFO
No
Defaults to 16C650 Init With 32-byte FIFO
Modified UART Type Detection Algorithm
Is EFR there?
No
Go to 16C550 Init With 16-byte FIFO
Yes
Is 850 type?
Yes
Go to 16C850 Init With 128-byte FIFO
No
Is FIFO size = 64?
Yes
Go to 16C654 Init With 64-byte FIFO
No
Is FIFO size = 32?
Yes
Go to 16C650 Init With 32-byte FIFO
No
Go to 16C550 Init With 16-byte FIFO

Figure 1A
Original UART Type Detection Algorithm

Figure 1B
Modified UART Type Detection Algorithm

2.2 PHOENIX BIOS DETECTION ERROR

The Phoenix BIOS version 4.0 release 6.0 uses a UART type detection algorithm (Figure 2A) that includes writing 0x00 to both DLL and DLM registers causing a detection error. When 0x00 is written into both DLL and DLM registers, the following read operations of DLL and DLM registers will return the value of DVID (0x02) and DREV (0x01), respectively, instead of the expected 0x00. If this algorithm is modified as shown in Figure 2B, this detection error can be eliminated. At the time of this publication, it has been observed that when a Windows O/S is used with this BIOS version, the Windows O/S performs its own detection of UARTs, thereby correcting the BIOS detection error, and the system operates properly.

FIGURE 2. ORIGINAL AND MODIFIED UART DETECTION SCHEMES IN PHOENIX BIOS VERSION 4.0 RELEASE 6.0

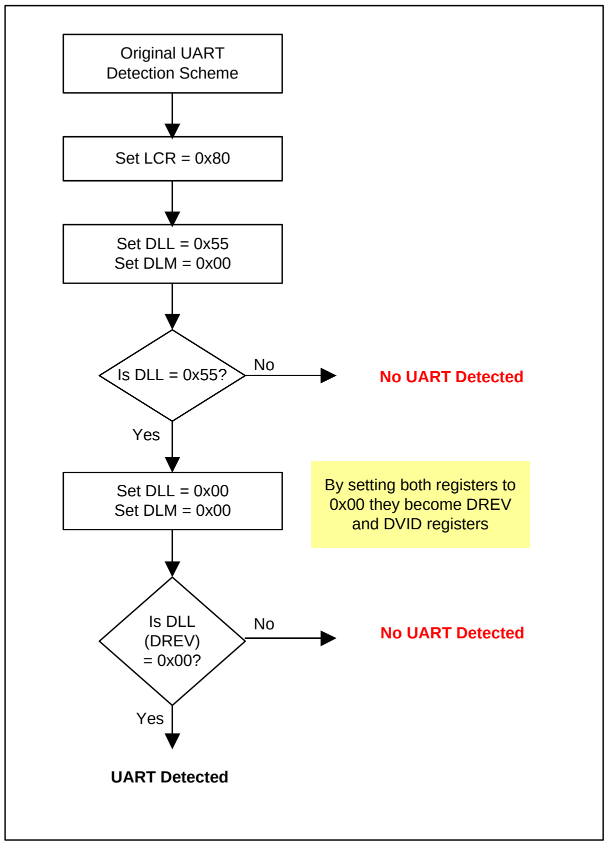

Figure 2A
Original UART Detection Scheme

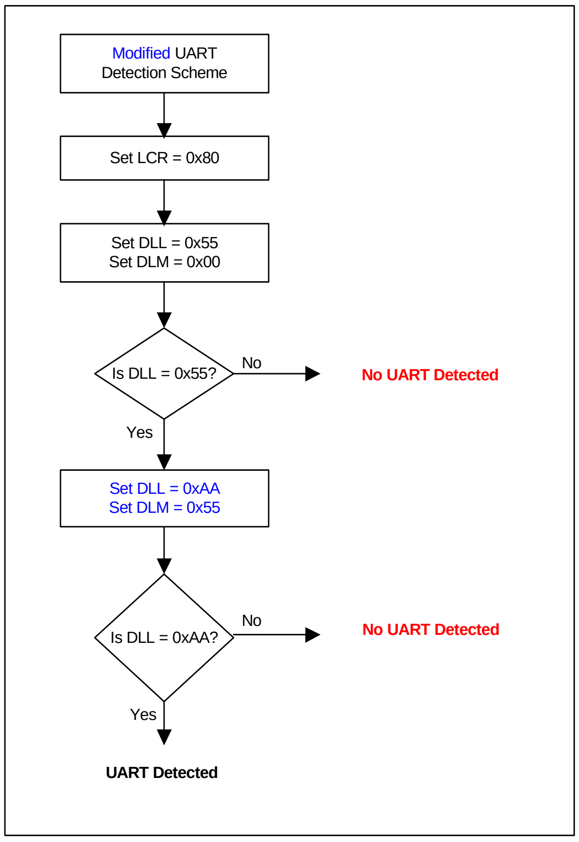

Figure 2B
Modified UART Detection Scheme

3.0 RECOMMENDED SOLUTIONS

In order to address the issues caused by the presence of the DVID, DREV and EFR registers, Exar is recommending the following solutions for the Linux O/S and the Phoenix BIOS. These solutions have been tested and reasonable effort has been made to ensure that these solutions work with the previous ST16C2550 version of “CC YYWW” as well as the current “A2 YYWW” part. These solutions will also be supported correctly in the upcoming revised device without the DVID, DREV and EFR registers. The upcoming XR16C2550 and ST16C2550, version “B2”, devices will be fully register compatible to the “CC YYWW” device. Both devices are identical except for the top marking (“XR” versus “ST”).

3.1 CASE 1 - LINUX 2.4.X OPERATING SYSTEM SOLUTIONS

There are two proposed solutions for Linux O/S kernel 2.4.x. as shown below. Solution A only requires the modification of the /etc/rc.serial file. Solution B, on the other hand, requires that the serial.c file be modified and the kernel to be recompiled.

SOLUTION A

This is a software patch for the Linux OS with kernel 2.4.x. If the file ‘/etc/rc.serial’ does not exist, create it and include the following lines in it:

```
setserial /dev/ttyS2 uart 16550A
setserial /dev/ttyS3 uart 16550A
```

Change the permissions of the rc.serial file by typing the following command:

```
chmod 755 rc.serial
```

In the system we've tested, the 2 channels of the ST16C2550 are ‘/dev/ttyS2’ and ‘/dev/ttyS3’ (for COM3 and COM4). These paths might be different in each system. Also, if the environment variable PATH is not set correctly, it may be necessary to explicitly specify the path for 'setserial'. In this case, replace the 'setserial' in the file ‘rc.serial’ by '/bin/setserial' or by '/sbin/setserial' depending on where the program ‘setserial’ resides in the system.

SOLUTION B

This is a software fix for the Linux 2.4.x kernel. This fix involves modifying the serial.c file and re-compiling the kernel by following the 18 steps given below. After this fix, the ST16C2550 “A2” device will be correctly identified as 16550A type. There is no need for the software patch mentioned in Solution A, which modifies the ‘/etc/rc.serial’ file.

Modify the ‘serial.c’ file located in this directory: /usr/src/linux/drivers/char

The original code looks like:

```
if (size_fifo(info) == 64)
   state->type = PORT_16654;
else
   state->type = PORT_16650V2;
```

The modified code should be:

```
if (size_fifo(info) == 64)
```

```
        state->type = PORT_16654;
    else if (size_fifo(info) == 32) {
        state->type = PORT_16650V2;
    else
        state->type = PORT_16550A;
    }
```

It is recommended that a copy of the original ‘serial.c’ file be saved for future reference. The kernel must be re-compiled using the steps given below:

1) Change to the directory ‘/usr/src/linux’. The directory 'linux' refers to the actual directory ‘linux-(kernel version)’

2) Back up the ‘.config’ file by moving it to another directory

3) Issue the command 'make mrproper'

4) Copy the default config file from the /usr/src/linux/configs directory to the current directory (/usr/src/linux). Rename it as ‘.config’

5) Run 'make xconfig' (if in X-windows), or else use 'make menuconfig'

6) There is no need to change any settings. Save and exit.

7) Edit the 'Makefile'. Append a suitable qualifier to the statement starting with EXTRAVERSION. This will ensure that the new kernel image will have a different name from the existing one.

8) Run 'make dep'

9) Run 'make clean'

10) Run 'make bzImage'

11) Run 'make modules'

12) Run 'make modules_install'

13) Run 'make install’

14) The newly created kernel image will be called ‘vmlinuz-(kernel version)-$EXTRAVERSION’ and will be copied to ‘/boot’ directory after the previous command.

15) Edit the file ‘/etc/lilo.conf’. Add another set of lines for the new kernel. If the new kernel does not start properly, you can always boot using the existing working kernel. In the example ‘lilo.conf’ file below, ‘/boot/vmlinuz-2.4.2-2’ is the original kernel image (label = linux) and ‘/boot/exar’ is the modified image (label = exar).

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
message=/boot/message
default=linux
timeout=50

image=/boot/vmlinuz-2.4.2-2
    label=linux
    read-only
    root=/dev/hda1
    append="console=tty console=ttyS0"
```

```
image=/boot/exar
   label=exar
   read-only
   root=/dev/hda1
   append="console=tty"
```

16) Save the '/etc/lilo.conf' file

17) Run 'lilo -v -v' to update the boot loader

18) Reboot the system and when the choice for kernels appear, select the newly created kernel.

3.2 CASE 2 - LINUX 2.6.x OPERATING SYSTEM SOLUTION

There is a proposed solution for Linux O/S kernel 2.6.x. as shown below. This solution involves modifying the '8250.c' file and re-compiling the kernel by following the 15 steps given below. After this fix, the ST16C2550 "A2" device will be correctly identified as 16550A type.

Modify the '8250.c' file which is located in this directory: '/usr/src/linux/drivers/serial'

The original code looks like:

```
if (size_fifo(up) == 64)
   up->port.type = PORT_16654;
else
   up->port.type = PORT_16650V2;
```

The modified code should be:

```
if (size_fifo(up) == 64)
   up->port.type = PORT_16654;
else if (size_fifo(up) == 32)
   up->port.type = PORT_16650V2;
```

It is recommended that a copy of the original '8250.c' file be saved for future reference. The kernel must be re-compiled using the steps given below:

1) Change to the directory '/usr/src/linux'. The directory 'linux' refers to the actual directory 'linux-(kernel version)'

2) Back up the '.config' file by moving it to another directory

3) Issue the command 'make mrproper'

4) Run 'make xconfig' (if in X-windows), or else use 'make menuconfig'

5) There is no need to change any settings. Save and exit.

6) Edit the 'Makefile'. Append a suitable qualifier to the statement starting with EXTRAVERSION. This will ensure that the new kernel image will have a different name from the existing one.

7) Run 'make dep'

8) Run 'make clean'

9) Run 'make bzImage'

10) Run 'make modules'

11) Run 'make modules_install'

12) Run 'make install'

13) The newly created kernel image will be called ‘vmlinuz-(kernel version)-$EXTRAVERSION’ and will be copied to ‘/boot’ directory after the previous command.

14) Edit the ‘/boot/grub/menu.1st’ file. Add another set of lines for the new kernel. If the new kernel does not start properly, you can always boot using the existing working kernel. In the example ‘menu.1st’ file below, ‘/boot/vmlinuz-2.6.4-52’ is the original kernel image (title Linux) and ‘/boot/exar’ is the modified image (title exar).

```
color white/blue black/light-gray
default 0
timeout 8
gfxmenu (hd0,3)/boot/message


###Don't change this comment - YaST2 identifier: Original name: linux###
title Linux
   kernel (hd0,3)/boot/vmlinuz root=/dev/hda4 vga=0x31a splash=silent desktop resume=/dev/hda3 showopts
   initrd (hd0,3)/boot/initrd.orig

###Don't change this comment - YaST2 identifier: Original name: exar###
title exar
   kernel (hd0,3)/boot/exar root=/dev/hda4 vga=0x31a splash=silent desktop resume=/dev/hda3 showopts
   initrd (hd0,3)/boot/initrd.exar

###Don't change this comment - YaST2 identifier: Original name: floppy###
title Floppy
   root (fd0)
   chainloader +1

###Don't change this comment - YaST2 identifier: Original name: failsafe###
title Failsafe
   kernel (hd0,3)/boot/vmlinuz root=/dev/hda4 showopts ide=nodma apm=off acpi=off vga=normal noresume nosmp noapic
maxcpus=0  3
   initrd (hd0,3)/boot/initrd

###Don't change this comment - YaST2 identifier: Original name: memtest86###
title Memory Test
   kernel (hd0,3)/boot/memtest.bin
```

15) Reboot the system and when the choice for kernels appear, select the newly created kernel.

Exar is still working on a software patch where the kernel does not require to be recompiled. Exar will advise customers as soon as the software patch is available.

3.3 CASE 3 - PHOENIX BIOS VERSION 4.0 RELEASE 6.0 SOLUTION

This BIOS version performs a unique sequence of writes and reads at I/O locations 0x3F8, 0x2F8, 0x3E8 and 0x2E8, which are the standard COM1 through COM4 locations. This BIOS determines if serial ports (UARTs) are actually present at these locations by checking the values returned by the reads. The sequence of steps that this BIOS performs to validate the existence of a serial port (UART) is given below:

1) Write 0x80 to LCR register; // Enable access to baud rate registers DLL & DLM in the shadow register page.
2) Write 0x55 to DLL;
3) Write 0x00 to DLM;
4) Read DLL; // Expect a value of 0x55. If data is correct, continue
5) Write 0x00 to DLL;
6) Write 0x00 to DLM; // DLL becomes DREV and DLM becomes DVID
7) Read DLL; // Expect a value of 0x00. Correct data validates the presence of a serial port.
   // "A2 YYWW" device fails this step because the value read is 0x01.
8) Write 0x00 to LCR; // Return to base register page.

Here is the sequence of steps to read the Device Revision and Device ID registers:

a) Write 0x80 to LCR; // Enable access to baud rate registers DLL and DLM in the shadow register page.
b) Write 0x00 to DLL;
c) Write 0x00 to DLM;
d) Read DLL; // This gives a value of 0x01 for device revision
e) Read DLM; // This gives a value of 0x02 indicating that the device is ST16C2550

Writing a value of 0x00 to both DLL and DLM registers is not a valid setting since the resulting baud rate is undetermined. The Device ID and Device Revision registers should not interfere in normal operation.

SOLUTION FOR PHOENIX BIOS VERSION 4.0 RELEASE 6.0 ISSUE

This problem can be resolved by modifying the Phoenix BIOS by writing alternate values to DLL and DLM in steps 5) and 6) above. This will detect the two serial ports of the "A2" device correctly. For example, change the steps 5), 6) and 7) above as follows:

5) Write 0xAA to DLL;
6) Write 0x55 to DLM; // DLL and DLM stay as DLL and DLM
7) Read DLL; // Expect a value of 0xAA. Correct data validates the presence of a serial port.
   // "A2 YYWW" device will pass this step correctly.

TABLE 1: SUMMARY OF AFFECTED PARTS

AFFECTED PARTS	LINUX KERNEL 2.4.X O/S	PHOENIX BIOS VER-4.0 REL-6.0	WINDOWS 98, NT4.0, CE, 2000 AND XP O/S	EMBEDDED SYSTEMS	IMMEDIATE COMPATIBLE PARTS FOR LINUX (NOTE 1)	NEW DIRECT REPLACEMENT PARTS FOR PHOENIX BIOS AND LINUX O/S (NOTE 2)
ST16C2450CJ44 ST16C2450IJ44 ST16C2450CP40 ST16C2450IP40 ST16C2450CQ48 ST16C2450IQ48	No Error	Error is caused by the wrong value returned from DLM or DLL register. See section “3.3” for a solution.	No Error.	Unlikely because a proprietary driver is highly used but we suggest you verify it.	XR16L2750CJ XR16L2750IJ (no PDIP package) XR16L2750CM XR16L2750IM	XR16C2450IJ XR16C2450IP XR16C2450IM
ST16C2550CJ44 ST16C2550IJ44 ST16C2550CP40 ST16C2550IP40 ST16C2550CQ48 ST16C2550IQ48	Error is caused by detection of the EFR register. See sections “3.1” and “3.2” for solutions.	Error is caused by the wrong value returned from DLM or DLL register. See section “3.3” for a solution.	No Error.	Unlikely because a proprietary driver is mostly used but we suggest you verify it.	XR16L2750CJ XR16L2750IJ (no PDIP package) XR16L2750CM XR16L2750IM	XR16C2550IJ XR16C2550IP XR16C2550IM
ST16C2552CJ44 ST16C2552IJ44	Error is caused by detection of the EFR register. See sections “3.1” and “3.2” for solutions.	Error is caused by the wrong value returned from DLM or DLL register. See section “3.3” for a solution.	No Error.	Unlikely because a proprietary driver is mostly used but we suggest you verify it.	XR16L2752CJ XR16L2752IJ	XR16C2552IJ

Note:

1. Under Linux operating environment the XR16L2750 may be a replacement part but at a higher price due to its 64-byte FIFO size.
2. New direct replacement parts will be offered in the industrial grade only and samples should be available by mid December-04.

4.0 Q&A

Q1 What will be the top marking for the upcoming XR16C2550 and ST16C2550, version “B2” devices?

A1 The upcoming XR16C2550 and ST16C2550, version “B2” devices will carry a top marking of “B2 YYWW” below the part number.

Q2 What is the difference between the package code suffix “Q” and “M”?

A2 For the upcoming XR16C2550 and ST16C2550, version “B2” devices , the package suffix “Q” and “M” refer to the same package. For example, ST16C2550CQ48 and XR16C2550IM are both housed in a 48-lead Thin Quad Flat Pack (7 x 7 x 1.0mm).

Q3 Is the previous “CC YYWW” device still available?

A3 No. “CC YYWW” parts are no longer available from Exar or our authorized distributors. We are now shipping “A2 YYWW” device.

Q4 Is this application affecting any of the Windows Operating Systems?

A4 The “A2” device has been verified to operate correctly with the following operating systems: Windows 98, NT4.0, CE, 2000 and XP.

Q5 Is there an immediate replacement device?

A5 Yes, for the Linux O/S, the XR16L2750 will be an immediate replacement part but it carries a higher price because of its larger FIFO size of 64-byte. See the Summary of Affected Parts Table (Table 1).

Q6 When will there be a direct replacement?

A6 Exar is expediting a new direct replacement part with samples available in mid-December, 2004. Please make your request for samples now. See the Summary of Affected Parts Table (Table 1).

Q7 Will the “A2” version become obsolete?

A7 Exar will continue to support customers who are not affected by issues mentioned in this application note, with the current “A2” version until the current inventory is no longer available. Thereafter, customers will be supported with the version “B2”.

Q8 If I implement the suggested software solution, will it work with previous versions such as “CC YYWW”, the current “A2” version and the new direct replacement version?

A8 Yes. These solutions have been tested and reasonable effort has been made to ensure that these solutions work with the previous ST16C2550 version of “CC YYWW” as well as the current “A2 YYWW” part. These solutions will also be supported correctly in the upcoming revised device without the DVID, DREV and EFR registers. The upcoming XR16C2550 and ST16C2550, version “B2”, devices will be fully register compatible with the “CC YYWW” device.

Q9 Is there a Linux O/S release version 2.5.x?

A9 No. Linux O/S (Kernel) releases are made on even number versions. Currently the Linux community is working on version 2.7.x (engineering version) to be released as version 2.8.x in the future.

Q10 Is there a software patch available for the latest released Linux 2.6.x kernel?

A10 Exar is working on a software patch and will advise as soon as it is available.