AN984
APPLICATION NOTE

HOW TO USE THE ST9 HDS2V2 EMULATOR WHEN DEVELOPING USB APPLICATIONS

by Microcontroller Division

INTRODUCTION

This application note provides certain guidelines on how to use the ST9 HDS2V2 EMULATOR when developing USB applications.

1 POWERING SEQUENCE

Be careful that no part of the system emulator-application board is powered up before the entire installation session has been performed.

The emulator should be powered up first and the target board should be powered up second in order to prevent the probe from being damaged. The power-down sequence is carried out in reverse order.

1.1 USB BUS POWERED APPLICATIONS

You should take a special care when developing bus-powered USB applications because in this case power comes from the USB cable. Do not forget to power on the emulator before plugging in the USB cable and to remove the USB cable before powering off the emulator.

2 USING BREAKPOINTS

If the program stops on a breakpoint while using USB application firmware on the emulator, communication with the USB target PC may be cutoff.

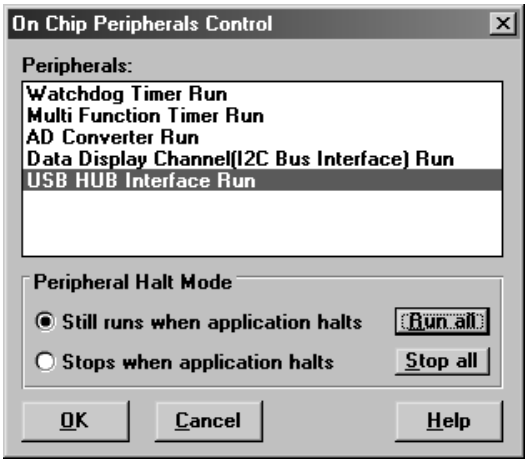
Some ST92163 peripherals can be frozen when the program is stopped by the debugger. By default, peripherals will still run if the program is stopped by the debugger.

For the ST92163 USB peripheral, when the program stops on a breakpoint and the peripheral remains running, only the NAK handshake can be performed. **No data transfers consecutive to a SETUP, OUT and IN tokens can be performed when the program is stopped on a breakpoint.**

To configure the peripheral behaviour, open the Command menu of the ST9+ debugger interface and select On-Chip Peripheral Control.

Figure 1. Command Menu

Figure 2. On-Chip Peripheral Control Menu



3 MEMORY MAPPING WITH EXTERNAL MEMORY INTERFACE

3.1 SYSTEM MEMORIES

ST9+ chip's internal memories, such as RAM and ROM, are devalidated for emulation purposes. In the emulator, they are replaced by emulation memories that are used to debug an application program by modifying the code whenever an error is detected.

These emulation memories are located on the emulator itself. They are called System Memories.

3.2 USER MEMORIES

Segments that identify external memories can be mapped in two different ways:

- using on-emulator memories to emulate those segments. In this case, these memories are defined as system memories,
- using memories located on your application board. In this case, these memories are defined as user memories.

In both cases, all emulation features such as breakpoints, trace, dump and memory access control functions are available.

Some behavioral differences must be taken into consideration:

- When mapping external segments as user memories, the emulator access time is slower than the chip access time (buffers add nearly 5 ns delay, depending on conditions). Refer to the probe schematics for more precise information.
- When mapping external segments as system memories, the following points must be taken into account:
  - When external memory is accessed through port0 and port1, the emulator does not detect if the ports are not correctly configured as alternate functions for accessing external memories. If this configuration has not been handled correctly, the application program is able run correctly with system memories, whereas it cannot run with user memories or with the chip itself.
  - The emulator will function at its maximum operating speed without requiring Wait states for accessing the system memories, whereas Wait states may be required by the chip for accessing application memories.

In order to avoid external memory interface configuration problems, it is recommended that the application program be debugged using the following two-step procedure:

- Step 1 consists of debugging the program with the emulation memories.
- Step 2 consists of debugging the program with the application memories to verify that ports and access configurations are correct.

3.3 MEMORY LOCATION

The ST92163 MCU is generally configured by memory segment.

As the MMU bus is not available as an alternate function, only one of the 1h, 2h, 3h, 22h or 23h segments can be chosen to be an application segment. Others are displayed by the emulator, but do not exist on the chip.

Table 1. ST92163 Memory Locations

ST92163 Memory Areas	Memory Types
RAM (segment 20h)	System
Segment 21h	Reserved
ROM (segment 0h, 1h)	System
Segments 2h, 3h, 22h, 23h	System or User

3.4 MAPPING THE MEMORY SPACE

The memory space can be mapped either by using a mapping file named `hardware.gdb` or by configuring memory spaces using the debugger tool. A standard memory mapping file, `hardware.gdb,` is provided with the Windows debugger.

Refer to the ST92163 HDS2V2 Emulator User Manual for more details on memory mapping space.

3.4.1 Mapping an External Memory

The following example describes how to map 128 Bytes of RAM as external memory.

The *hardware.gdb* file must modified as follows:

Figure 3. hardware.gdb File

```
clear_map

map ORIGIN=0x000000  LENGTH=64K    TYPE=SR
map ORIGIN=0x034040  LENGTH=128    TYPE=UW
map ORIGIN=0x20f800  LENGTH=2K     TYPE=SW
```

The various types of mapping modes are specified as follows:

sw: meaning system emulated memory, all access types (read and write)allowed

sr: meaning read-only emulated memory, read access only

uw: meaning user memory, all access types (read and write) allowed

You can declare an array in the *ext_ram.c* file:

Figure 4. ext_ram.c File

```
#include "extram.h"
unsigned char ExternalRAM[128];
```

The associated *ext_ram.h* file will be as follows:

Figure 5. ext_ram.c File

```
extern unsigned char ExternalRAM[128];
```

The *ext_ram.scr* script file has to be modified as follows,:

Figure 6. ext_ram.scr File

```
MEMORY
{
   ROM : ORIGIN = 0x000000, LENGTH = 20K,
   RAM : ORIGIN = 0x20F800, LENGTH = 2K,
   EXTERNAL_RAM : ORIGIN = 0x34040, LENGTH = 128
}
...
...
SECTIONS
...
...
.data :
  {
   _extram_start = .;
      extram.o(.bss) _extram_end = .;
   } > EXTERNAL_RAM
```

NOTE: The length is given in BYTES. If you want to map kilobytes of memory space, use the K suffix. 1K stands for 1024 bytes

4 DEBUGGER AND COMPILATION

The .u file must be closed in the ST9+ debugger interface before the project is compiled, otherwise the following error message will be displayed:

```
D:\TOOLSS~1\BIN/ld9.exe problem opening output file USBcore.u, Permission denied (EACCES)(gmake) gmake.exe: *** [USBcore.u] Error 1
```

5 REFERENCE DOCUMENTS

1. ST92163 HDS2V2 Emulator User Manual, STMicroelectronics, Release 2.2, August 1999

2. “Application Note AN1069: Addressing up To 4 Mbytes of Memory from an ST9+ with a16-bit External Bus”, STMicroelectronics Microcontroller Division Applications, July 2000

3. ST92163 Datasheet, STMicroelectronics, Release 1.9, January 2000

"THE PRESENT NOTE WHICH IS FOR GUIDANCE ONLY AIMS AT PROVIDING CUSTOMERS WITH INFORMATION REGARDING THEIR PRODUCTS IN ORDER FOR THEM TO SAVE TIME. AS A RESULT, STMICROELECTRONICS SHALL NOT BE HELD LIABLE FOR ANY DIRECT, INDIRECT OR CONSEQUENTIAL DAMAGES WITH RESPECT TO ANY CLAIMS ARISING FROM THE CONTENT OF SUCH A NOTE AND/OR THE USE MADE BY CUSTOMERS OF THE INFORMATION CONTAINED HEREIN IN CONNEXION WITH THEIR PRODUCTS."

littlediode
COMPONENTS FOR ALL

Littlediode supplies new, hard to find or obsolete electronic components and semiconductors all over the world.

With over two million different components listed you are sure to find the part you need.

Feel free to visit us today at our online store:

LittleDiode.com

Looking forward to providing you with the best possible service.