Application Note AN212

Using 512k Flash Memory & 512k SRAM in Rabbit-Based Systems

Introduction

The Rabbit 2000™ microprocessor can address up to one megabyte (1M) of memory. In most implementations, this memory is split between flash memory and static RAM, usually in increments of 128k, 256k or 512k. (The amounts of flash memory and SRAM need not be equal.) When using Dynamic C versions 7.03 and 7.04 (and earlier versions), there are limitations on using more than 256k of either memory type. This Application Note and the accompanying ZIP file describe techniques for making use of up to 512k of flash memory or SRAM (in 2 x 256k chip configuration) under these versions of Dynamic C.

(Note: These issues will be addressed in Dynamic C 7.05 so that making full use of the 1M address space will be nearly transparent to the user.)

Using a Second Flash Memory

Some Z-World/Rabbit Semiconductor controller boards come with two 256k flash memory chips. User-designed systems may use similar memory configurations.Under Dynamic C 7.03/7.04, the second flash chip can be used for the Flash File System (FFS), but a small change is needed if more than 256k of RAM is to be used in the same system. Versions of Dynamic C prior to 7.05 cannot take advantage of the second flash for code without some BIOS and library code changes. The second flash on Z-World boards is designated by default to be used for the Flash File System.

For Dynamic C 7.05 and later, the only change required to use the second flash for program code will be to uncomment out the following line in the BIOS:

```
//#define USE_2NDFLASH_CODE
```

The methods and changes described were developed and tested under version 7.03 and 7.04 only, but it is possible to retrofit the changes to earlier versions. Dynamic C 7.04 has some of the changes here incorporated into it, so fewer changes are required where noted.

Here are the ways two 256k flash chips can be used with 512k or less RAM:

- Use the 2nd flash for the Flash File System, use the extra RAM for **xalloc()**.
- Use the 2nd flash for xmem code and constants, use the extra RAM for **xalloc()**.
- Use 2 coresident programs (per Application Note 210).

For example, the RabbitCore 2000 comes with 512k of RAM and 256k of flash memory. The changes to BIOS and library files given here will make the full 512K available to the progam.

Each of these usages has a section below discussing it. A user not interested in the details of what changes are needed and why can simply copy the files needed for the chosen configuration into the appropriate places and make any other required changes. Users who want details of the changes made to facilitate a memory configuration can read the explanations after the listed changes. Note that using the Flash File System is not compatible with either using the second flash for code/constants, or using the second flash for a second coresident program.

APNOTE212.ZIP contains all of the files referenced in the discussions except for **DEFAULT.H**, which requires a small change depending on which board is used. It is recommended to save old copies of files that will be replaced with like-named files by changing their extensions to **.BAK**. Here are the files contained in **APNOTE212.ZIP**:

- **APNOTE212BIOS.C** - A special BIOS to use for all the memory usages discussed here except compiling two co-resident programs. Use Options | Compiler in Dynamic C to specify this as the user defined BIOS.
- **FLASHWR.LIB** - A replacement for the like named file in **\LIB\BIOSLIB**. This version ensures that the flash transfer buffer is allocated at the top of used RAM for all of the usages described in this document. It also allows the System ID block to be overwritten when using the second flash for code.
- **STACK.LIB** - A replacement for the like named file in \LIB\BIOSLIB\ . This version adjusts the initialization of the extended memory so that the correct areas of RAM will be available for all of the usages described in this document.
- **WRSYSID.C** - A program to rewrite the System ID Block if desired.
- **FLASH.LIB**, **FS_FLASH.LIB** - Replacements for the like named files in **\LIB\FILESYSTEM**. These are changed to reflect memory mapping changes made in **APNOTE212BIOS.C**

Copy all of these files to the apppropriate location. (Read the appropriate usage section first, not all of the usages discussed require changes.)

Using a Second Flash Memory for FFS

To use the second flash memory for the Flash File System, no changes are required for Dynamic C 7.03 or 7.04 if the RAM size is less than 512K. If 512K of RAM is used, no changes are required for 7.04, but for 7.03, use the files accompanying this document, and make sure that the macro definition:

```
//#define USE_2NDFLASH_CODE
```

is commented out.

For Dynamic C 7.03, **STACK.LIB** is needed to correctly initialize RAM xmem allocation so that all unused portions of the RAM are available for **xalloc()** when the RAM size is 512K. A change in the quadrant mapping in the BIOS requires changes to the second flash driver *only* in 7.03.

Using a Second Flash Memory for xmem Code/Constants

To use the second flash memory for xmem code and constants, make sure that the macro definition:

```
#define USE_2NDFLASH_CODE
```

is *not* commented out.

Additional change needed:

The following lines of code in **\LIB\DEFAULT.H** are used to default the board type to the JackRabbit board if no System ID block is detected:

```
#if (_BOARD_TYPE_ == 0x0000)
#undef  _BOARD_TYPE_
#define _BOARD_TYPE_ BL1810
#endif
```

For boards other than the JackRabbit, that entire block of code should be commented out and replaced by:

```
#undef  _BOARD_TYPE_
```

followed by *one* of these definitions:

```
#define _BOARD_TYPE_OP6600 // 18MHz Intellicom, no Ethernet
#define _BOARD_TYPE_RTDK   // 18MHz TCP/IP Toolkit
#define _BOARD_TYPE_OP6700 // 18MHz Intellicom + Ethernet
#define _BOARD_TYPE_ EG2000  // 22MHz EtherGate,2 enet ports
#define _BOARD_TYPE_ RABLINK // 22MHz RabbitLink
#define _BOARD_TYPE_ EG2100  // 22MHz RabbitLink
#define _BOARD_TYPE_ EG2020  // 22MHz EtherGate,one enet port
#define _BOARD_TYPE_ RCM2100 // 22MHz ECM,enet,512K/512K
#define _BOARD_TYPE_ RCM2110 // 22MHz ECM,ethernet,256K/128K
#define _BOARD_TYPE_ RCM2120 // 22MHz AECM,512K/512K
```

```
#define _BOARD_TYPE_ RCM2130 // 22MHz AECM,256K/128K
#define _BOARD_TYPE_ RCM2115 // 22MHz ECM,ethernet,256K/128K
```

Explanation:

For 7.03, **STACK.LIB** is needed to initialize RAM **xmem** allocation correctly when RAM size is 512k. Because Dynamic C currently only detects the primary flash on CS0, **APNOTE212BIOS.C** changes

```
#define FLASH_SIZE _FLASH_SIZE_
```

to:

```
#define FLASH_SIZE _FLASH_SIZE_*2
```

when using the second flash for code, so that the compiler knows there is extra space for xmem code and constants. The BIOS also maps the second physical memory quadrant (40000h-7FFFFh) to CS2/OE0/WE0 so that Dynamic C sees the bottom 512k as a contiguous block of available flash. Another small change to the BIOS file was necessary to allow Dynamic C to write code to the second flash when downloading code or writing breakpoints. The code in **FLASHWR.LIB** that prevents the System ID block from being written is disabled in this special version.

When Dynamic C boots up a target board, the Board ID contained in the System ID Block is used to define the macro **_BOARD_TYPE_** which is used at compile time to determine which board specific libraries to use by default, among other things. Because Dynamic C prior to version 7.05 does not know how to compile "around" the system ID block located in the top of the primary flash, it will write over that block. The change to **DEFAULT.H** replaces the board type information lost when the System ID Block is overwritten.

WRSYSID.C is a program that can be used to restore the System ID Block if desired.

Using Two Coresident Programs

Application Note 210 and its accompanying ZIP file explain the changes needed to use the second flash memory for two coresident programs.

Using the RabbitCore Module RCM2000 with 512K of RAM

To use the full 512k of SRAM on the RCM2000 and similar boards, make sure that the macro definition:

```
//#define USE_2NDFLASH_CODE
```

stays commented out.

Explanation:

For Dynamic C 7.04, this will work "out of the box," without the files accompanying this Application Note. No changes are required. For Dynamic C 7.03, **STACK.LIB** is needed to correctly initialize RAM xmem allocation so that all unused portions of the RAM are available for **xalloc()**. Keeping the macro commented out prevents the BIOS from assuming two flashes are present.

Using Extra SRAM Space

For more about how the Rabbit Memory Management and Memory Interface Units (MMU,MIU) work, see Application Note 202.

The normal memory configuration that the Dynamic C BIOS sets up is as follows:

- 0000h - 5FFFh "root" code and constants
- 6000h - CFFFh "root" data.
- D000h - DFFFh stack segment
- E000h - FFFFh extended segment

The user can lower the root code-data boundary at 6000h to as low as 3000h by changing the **DATAORG** macro in the top of the BIOS, but the realistic limitation of root RAM available to the user program for variables is D000h-3000h, or 40960d bytes. Until such time as far pointers are implemented in Dynamic C, this limits the use of additional RAM space to allocating blocks for extra stack space or data storage or temporary extended memory buffers. The **xalloc()** function can be called by the user to allocate blocks of extra RAM for data storage. The uC/OS-II function **OSTaskCreate()** takes a stack size parameter and uses the **xalloc()** function internally. Segment registers used for the stack and extended memory segments make those segment able to address anywhere in the 1M physical address space. The Dynamic C functions **xmem2root, root2xmem,** and **xmem2xmem** allow manipulation of data in extended memory.

Another way to make use of of extra RAM is to compile the user program to RAM (use the Compiler Options dialog box) during development. This can be useful if it is critical that interrupts not be turned off for too long when debugging. Single-stepping and setting breakpoints in code involves writing to the code to change instructions into RSTs and then change them back again. If the code is in flash, then this causes interrupts to be turned off for as long 20 milliseconds depending on the flash device.

Hardware Issues

Using the second flash memory for code with Dynamic 7.03 or 7.04 (or earlier versions) will cause the System ID Block to be overwritten. The only information in the that block that is relevant to Dynamic C is the board ID and in the case of some Ethernet-enabled boards, the MAC address. The change discussed above to hardcode the board ID macro in **DEFAULT.H** takes care of the board ID issue for all boards.

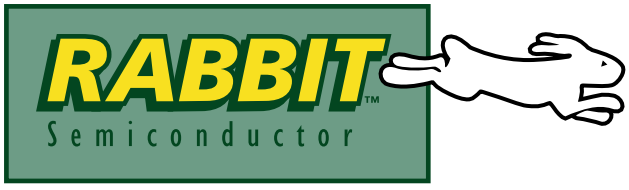
The MAC address of the TCP/IP Development Board is stored in a small EEPROM, and its IP drivers reads it from there, so the System ID block need not be duplicated in the second flash. The change to **DEFAULT.H** to unconditionally use the correct library will still be required.

The MAC address for the RabbitLink and RCM2100 is stored in the System ID block. The RabbitLink was released primarily to use canned software to allow Dynamic C to program and debug targets over the Internet, however it is programable by Dynamic C, so users

considering using the second flash for code with Dynamic C 7.03 or 7.04 and thereby overwriting the System ID Block need to consider this. If Dynamic C 7.05 is used, this not an issue.

If it is necessary to use the second flash for code before the release of 7.05 with a RabbitLink or RCM2100 board, then the **WRSYSID.C** program should be used to copy the system ID block to the top of the second flash. The BIOS files provided will ensure that it gets read from there.

To avoid backwards compatibility problems, it is best not to deploy RCM2100 and RabbitLink field upgradable units with the System ID Block only in the second flash. Users who will deploy field upgradable units are advised to upgrade to Dynamic C 7.05 and make sure any units developed with the System ID block overwritten in the first flash replace the block before deployment with **WRSYSID.C**.



Rabbit Semiconductor
2932 Spafford Street
Davis, California 95616-6800
USA

Tel. (530)757-8400
FAX (530)757-8402

Web site: http://www.rabbitsemiconductor.com