Enabling HighMem on the Malta™-R Development Board

Document Number: MD00958
Revision 01.01
July 26, 2013

Contents

Introduction

This Applications Note describes the implementation of HighMem for Linux on the MIPS® Malta™-R Development Platform. In most respects, this description is applicable to other MIPS-based platforms.

In Release 2 and lower of the MIPS architecture, Linux system memory is limited to the 512 MB in kseg0/1, which is mapped directly to physical memory (not via the TLB/FM). HighMem is a technique that can be used to enable the OS to access memory outside of kseg0/1. More information on HighMem can be found at http://linux-mm.org/HighMemory.

In Release 3 of the MIPS architecture, the size and attributes of memory segments can be programmed, thus avoiding this limitation.

1.1 Tools

This application note uses the following hardware and software tools:

- Linux kernel
  http://www.linux-mips.org/pub/linux/mips/mti-stable/v2.6/linux-mti-2.6.35.9-2.tar.gz

- Compiler
  4.5.2 (Sourcery CodeBench Lite 2011.03-110)
  https://sourcery.mentor.com/GNUToolchain/release1967

- Bit Files
  A00206-74Kc_2_3_0_i32d32t32HW-REF00306.fl
  A00206-74Kc_2_3_0-i32d32t32-REF00305.fl
  A00206-34Kc_2.5a_0-64ID-64TLB-100MHz-delay_REF00449.fl

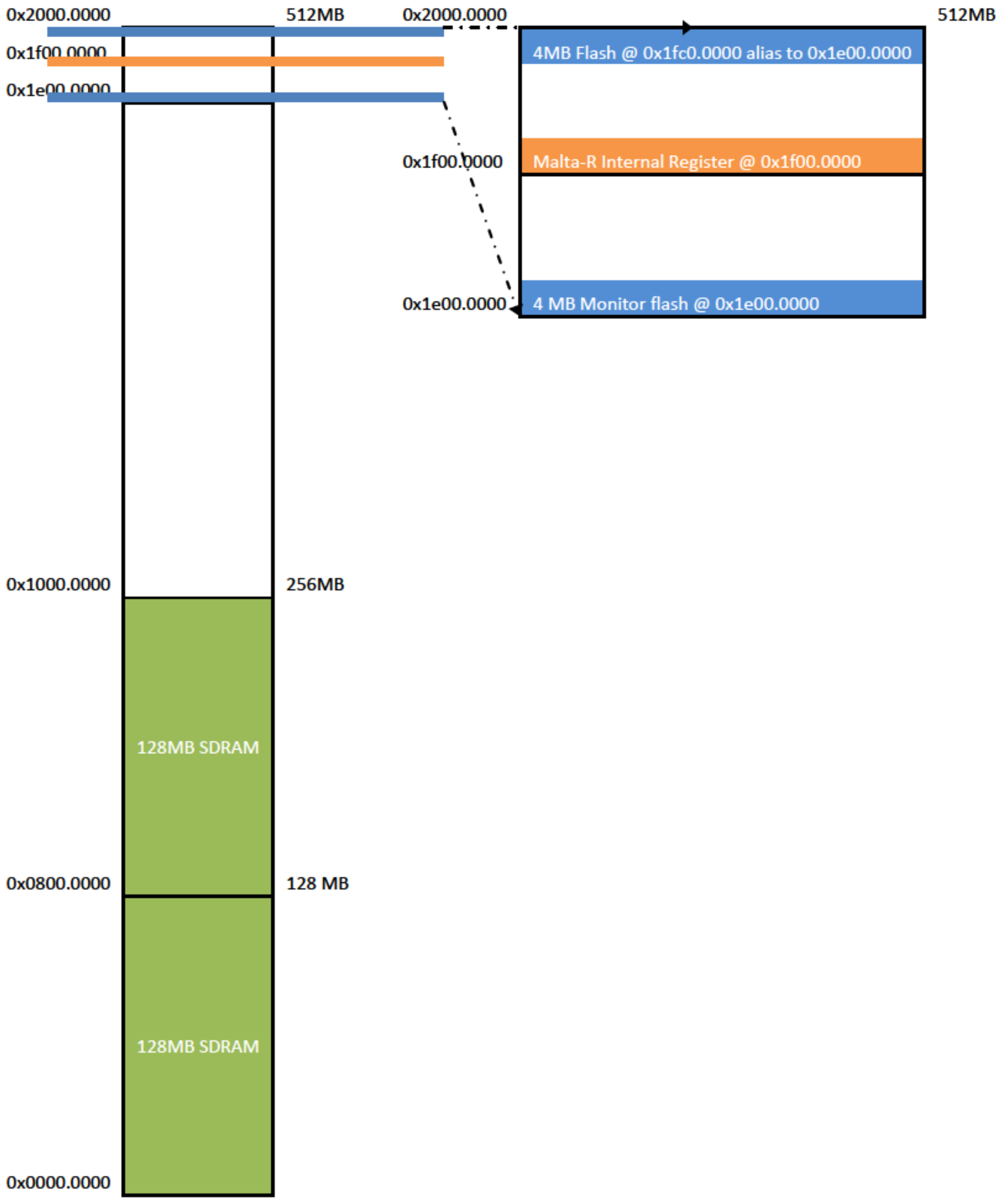
- Malta™-R Development Board with 256 MB RAM

- **busybox-1.19.3.tar.bz2**
  http://www.busybox.net/downloads/

- **Ubuntu 10 on a 32-bit host machine is used to compile the kernel and the busybox**

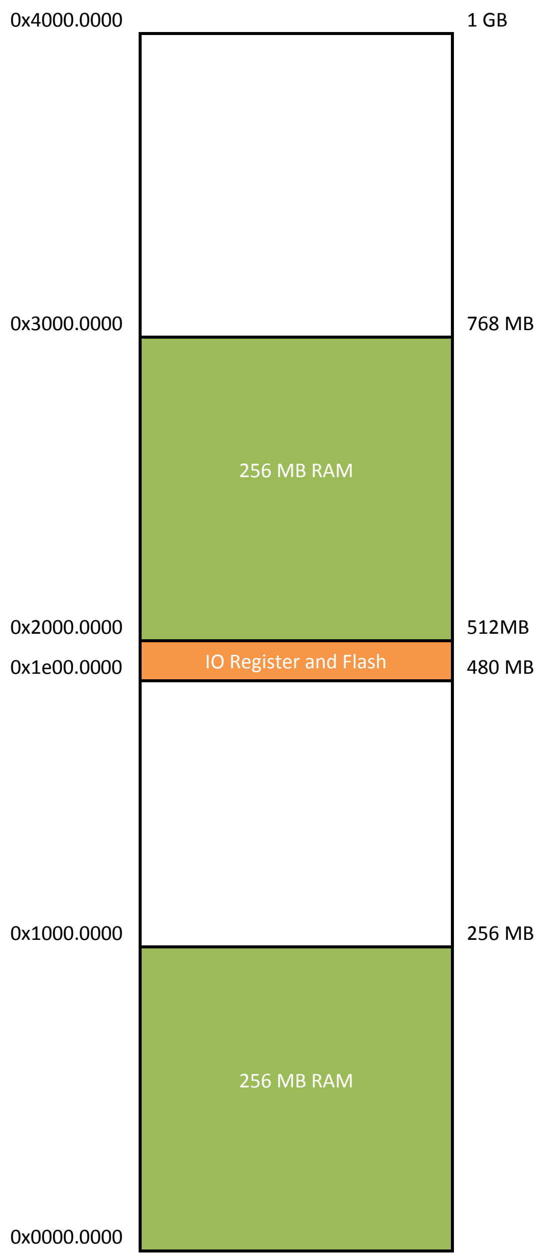
1.2 Malta™ -R Memory Map

Shown below is a high-level view of the address map for the MIPS® Malta™-R Development Platform with 256 Kbytes of RAM.



To support more than 512 MB of memory, a second bank of RAM must be added. For example, to support 512 MB, 2x256 MB RAM can be added, covering the physical RAM addresses from 0x0000.0000 and 0x2000.0000 respectively.

Note that in the MIPS architecture, kseg0 and kseg1 are mapped to the same physical address, and that the platform internal IO register must be in kseg0. Thus the maximum available memory is smaller than 512 MB.



For Linux to access 256 MB at physical address 0x2000.0000, HighMem is needed. To accomplish this, two steps are required:

- Enable HighMem in the Linux config file.
- Register 256 MB of memory for use as HighMem by Linux.

For step 1, the file arch/mips/Kconfig needs to be modified as shown below.

```
--- linux-mti-2.6.35.9-2/arch/mips/Kconfig      2011-09-20 14:43:08.000000000 -0700
+++ all/arch/mips/Kconfig       2012-06-25 13:10:12.506673553 -0700
@@ -227,6 +227,7 @@
        select SYS_SUPPORTS_SMARTMIPS
        select SYS_SUPPORTS_MICROMIPS
        select SYS_SUPPORTS_ZBOOT
+       select SYS_SUPPORTS_HIGHMEM
        help
          This enables support for the MIPS Technologies Malta evaluation
          board.
```

Then use “make menuconfig” to enable high memory support with a flat memory model:

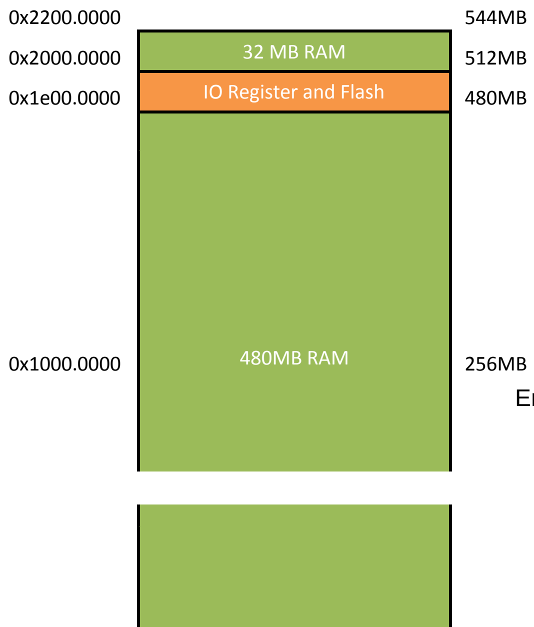
Select Kernel Type -> Enable High Memory Support ->Memory model (Flat Memory)

For step 2, simply add the following to the platform init routine:

```
add_memory_region(0x20000000, 0x10000000, BOOT_MEM_RAM);
```

This will register the second 256 MB RAM with the Linux kernel. Because the physical address is outside KSEG0, the kernel is hardcoded, such that it will mark it as high memory.

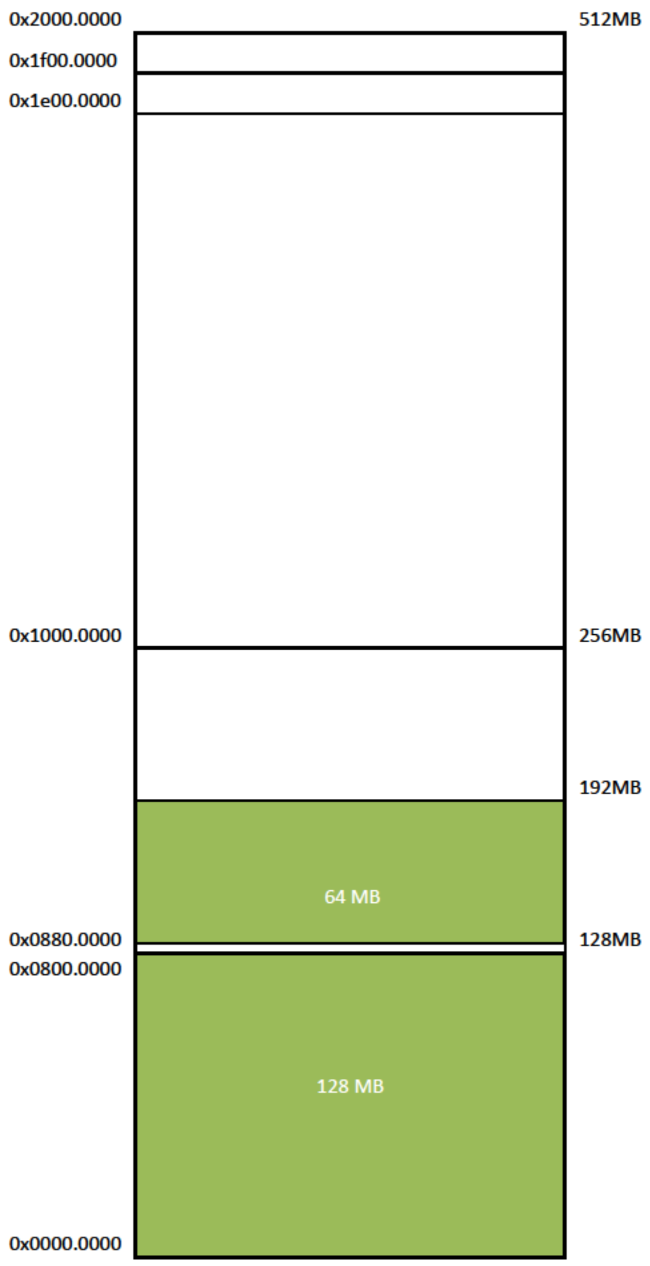
With this scheme, there is a 256 MB hole between two memory banks. Because a flat memory model is used, page table entries are created for the 256 MB hole. Assuming a 4 KB Linux page size, the total number of 32-byte entries for this hole are 256 MB/4 KB = 65536. The total memory wasted here is 65536*32=2 MB, so when using a flat memory model, it’s best to keep the hole as small as possible. One possibility is the remapping shown below.

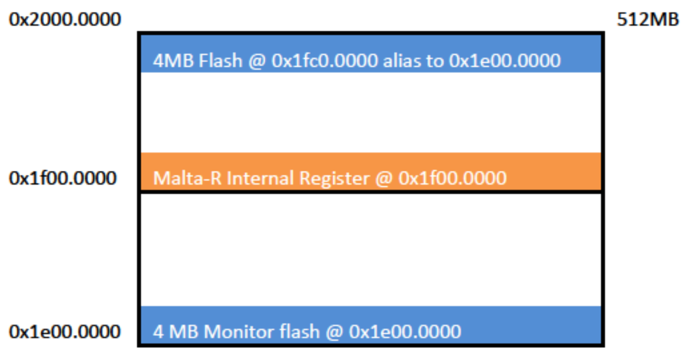


1.3 Malta™ -R Memory Map Emulating HighMem

The Malta Development Board does not support a second memory bank outside kseg0/kseg1, so the following changes are required to run HighMem on the board:

- Limit the kernel's use to the first 128 MB RAM for normal memory.
- Register the second bank of 64-MB RAM to start at 0x0880.0000, which leaves an 8-MB hole between the first and second memory banks.
- Hardcode HighMem to start after 0x0800.0000 (default is 0x1000.0000), such that the second memory bank will be used as high memory.




0x2000.0000
512MB
4MB Flash @ 0x1fc0.0000 alias to 0x1e00.0000
0x1f00.0000
Malta-R Internal Register @ 0x1f00.0000
0x1e00.0000
4 MB Monitor flash @ 0x1e00.0000

2 Building the Linux Kernel for the Malta™-R Development Board

To enable HighMem on the Malta board, the following steps are required:

1. Patch the Linux kernel for HighMem bug fixed
2. Configure Linux to support HighMem
3. Limit usable memory to 128
4. Register the memory used as HighMem

Step 3 is specific to the Malta board; steps 1, 2, and 4 apply to all other boards.

2.1 Patch the Linux Kernel for HighMem bug fixed

The Linux kernel posted on http://www.linux-mips.org/pub/linux/mips/mti-stable/v2.6/linux-mti-2.6.35.9-2.tar.gz does not have the latest HighMem bug fixed. To use this version, the patch must be applied as shown below.

```
$ tar -xzf linux-mti-2.6.35.9-2.tar.gz
$ cd linux-mti-2.6.35.9-2
$ patch -p1 < ../highmem.patch
patching file arch/mips/include/asm/cacheflush.h
patching file arch/mips/include/asm/cpu-features.h
patching file arch/mips/include/asm/fixmap.h
patching file arch/mips/include/asm/highmem.h
patching file arch/mips/include/asm/mach-generic/spaces.h
patching file arch/mips/include/asm/mipsregs.h
patching file arch/mips/Kconfig
patching file arch/mips/kernel/setup.c
patching file arch/mips/mm/cache c
patching file arch/mips/mm/c-r4k.c
patching file arch/mips/mm/dma-default.c
patching file arch/mips/mm/fault.c
patching file arch/mips/mm/highmem.c
patching file arch/mips/mm/init.c
patching file arch/mips/mm/sc-mips.c
patching file arch/mips/mti-malta/malta-init.c
patching file arch/mips/mti-malta/malta-memory.c
patching file arch/mips/mti-malta/malta-setup.c
patching file drivers/ide/ide-taskfile.c
patching file mm/highmem.c
patching file mm/memory.c
```

If the git repository is used, the patch is not required, because the HighMem patch has been submitted to the server.

```
$ git clone git://git.linux-mips.org/pub/scm/linux-mti.git -b linux-mti-2.6.35.9
Initialized empty Git repository in /home/keng/local/project/linux/git/test/linux-mti/.git/
remote: Counting objects: 3074346, done.
remote: Compressing objects: 100% (472645/472645), done.
remote: Total 3074346 (delta 2605148), reused 3043473 (delta 2574420)
Receiving objects: 100% (3074346/3074346), 613.56 MiB | 1.80 MiB/s, done.
Resolving deltas: 100% (2605148/2605148), done.
```

or

```
$ git clone git://git.linux-mips.org/pub/scm/linux-mti.git
$ cd linux-mti
$ git checkout linux-mti-2.6.35.9

Checking out files: 100% (36758/36758), done.
Branch linux-mti-2.6.35.9 set up to track remote branch linux-mti-2.6.35.9 from origin.
Switched to a new branch 'linux-mti-2.6.35.9'
```

2.2 Configure Linux to Support HighMem

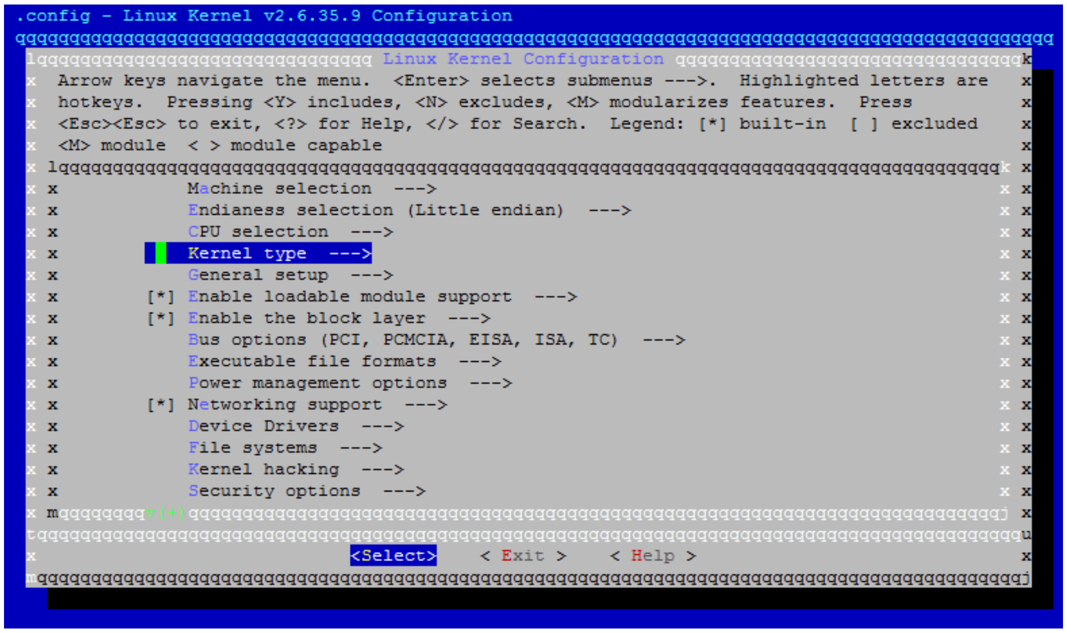
To enable HighMem in Linux, “select SYS_SUPPORTS_HIGHMEM” must be added to the file arch/mips/Kconfig.

```
--- linux-mti-2.6.35.9-2/arch/mips/Kconfig      2011-09-20 14:43:08.000000000 -0700
+++ all/arch/mips/Kconfig       2012-06-25 13:10:12.506673553 -0700
@@ -227,6 +227,7 @@
        select SYS_SUPPORTS_SMARTMIPS
        select SYS_SUPPORTS_MICROMIPS
        select SYS_SUPPORTS_ZBOOT
+       select SYS_SUPPORTS_HIGHMEM
        help
          This enables support for the MIPS Technologies Malta evaluation
          board.
```

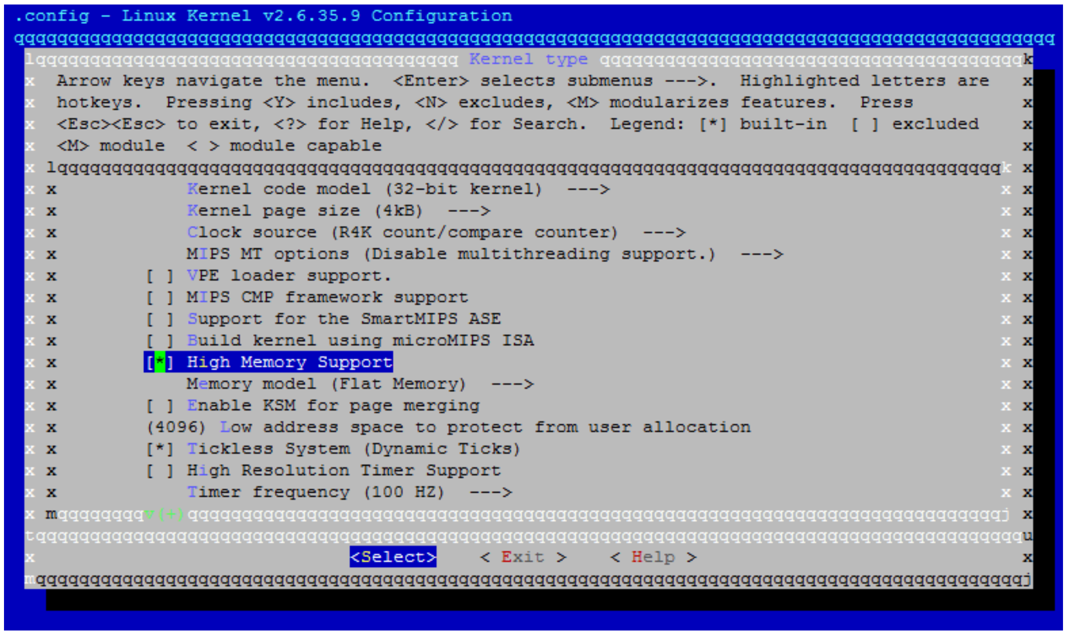
The next step, shown below, is to copy the default config file and run make to configure the kernel.

```
$ cp arch/mips/configs/maltaup_defconfig .config
$ make CROSS_COMPILE=mips-linux-gnu- menuconfig
```

Then select Kernel Type and enable High Memory Support.



```
.config - Linux Kernel v2.6.35.9 Configuration
                          Linux Kernel Configuration
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are
  hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press
  <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ] excluded
  <M> module  < > module capable

                Machine selection  --->
                Endianess selection (Little endian)  --->
                CPU selection  --->
                Kernel type  --->
                General setup  --->
            [*] Enable loadable module support  --->
            [*] Enable the block layer  --->
                Bus options (PCI, PCMCIA, EISA, ISA, TC)  --->
                Executable file formats  --->
                Power management options  --->
            [*] Networking support  --->
                Device Drivers  --->
                File systems  --->
                Kernel hacking  --->
                Security options  --->
          v(+)

                      <Select>    < Exit >    < Help >
```

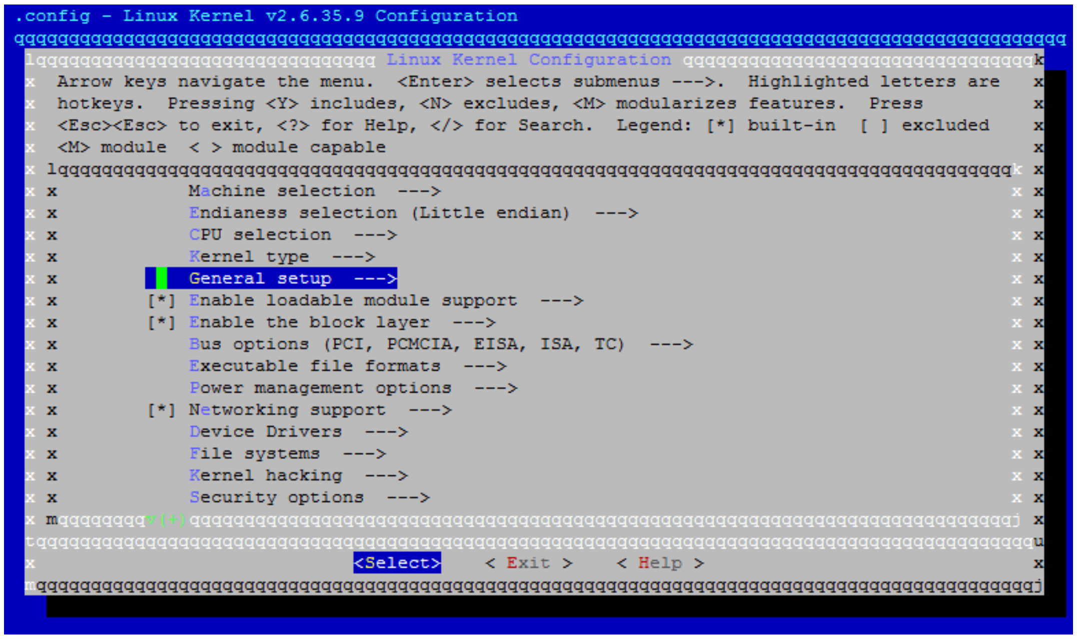


```
.config - Linux Kernel v2.6.35.9 Configuration
                                  Kernel type
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are
  hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press
  <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ] excluded
  <M> module  < > module capable

                Kernel code model (32-bit kernel)  --->
                Kernel page size (4kB)  --->
                Clock source (R4K count/compare counter)  --->
                MIPS MT options (Disable multithreading support.)  --->
            [ ] VPE loader support.
            [ ] MIPS CMP framework support
            [ ] Support for the SmartMIPS ASE
            [ ] Build kernel using microMIPS ISA
            [*] High Memory Support
                Memory model (Flat Memory)  --->
            [ ] Enable KSM for page merging
            (4096) Low address space to protect from user allocation
            [*] Tickless System (Dynamic Ticks)
            [ ] High Resolution Timer Support
                Timer frequency (100 HZ)  --->
          v(+)

                      <Select>    < Exit >    < Help >
```

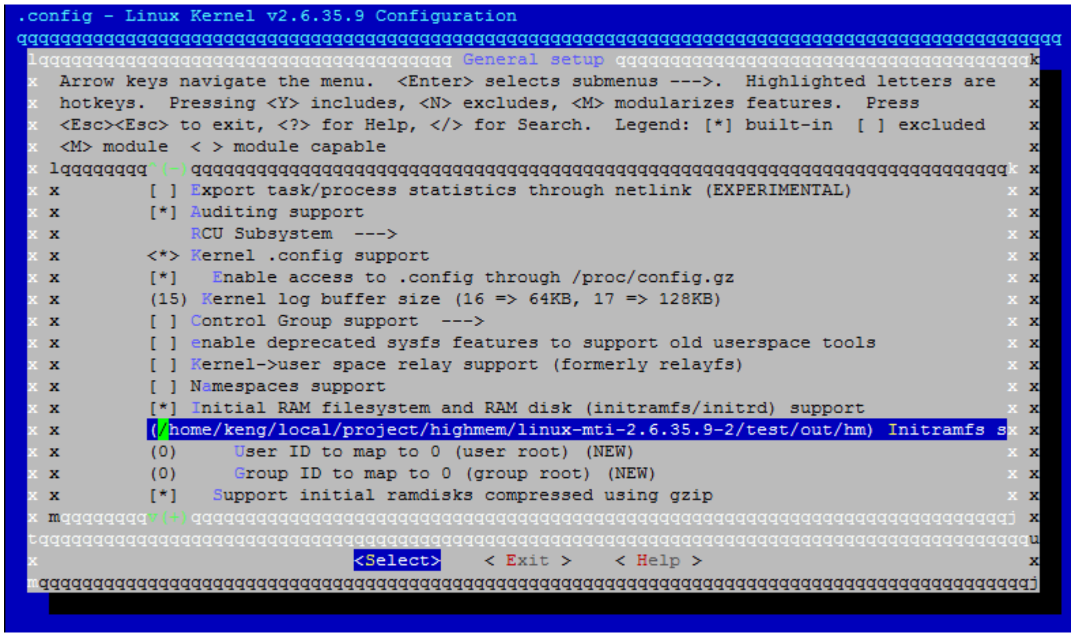
In this example, a Flat Memory model is used with the busybox root file system. It assumes that the root file system is installed in the following directory (and make sure that the proper /dev/* and /init files are created):

```
~/local/project/highmem/linux-mti-2.6.35.9-2/test/out/hm
```

Then do the following in order to configure the kernel to get the ramfs from the above directory:



```
.config - Linux Kernel v2.6.35.9 Configuration
 lqqqqqqqqqqqqqqqqqqqqqqqqqqqqqq Linux Kernel Configuration qqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqk
 x  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are  x
 x  hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press         x
 x  <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ] excluded   x
 x  <M> module  < > module capable                                                          x
 x lqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqk x
 x x             Machine selection  --->                                                  x x
 x x             Endianess selection (Little endian)  --->                                x x
 x x             CPU selection  --->                                                      x x
 x x             Kernel type  --->                                                        x x
 x x             General setup  --->                                                      x x
 x x         [*] Enable loadable module support  --->                                     x x
 x x         [*] Enable the block layer  --->                                             x x
 x x             Bus options (PCI, PCMCIA, EISA, ISA, TC)  --->                           x x
 x x             Executable file formats  --->                                            x x
 x x             Power management options  --->                                           x x
 x x         [*] Networking support  --->                                                 x x
 x x             Device Drivers  --->                                                     x x
 x x             File systems  --->                                                       x x
 x x             Kernel hacking  --->                                                     x x
 x x             Security options  --->                                                   x x
 x mqqqqqqqqv(+)qqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqj x
 tqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqu
 x                             <Select>    < Exit >    < Help >                             x
 mqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqj
```



```
.config - Linux Kernel v2.6.35.9 Configuration
 lqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqq General setup qqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqk
 x  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are  x
 x  hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press         x
 x  <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ] excluded   x
 x  <M> module  < > module capable                                                          x
 x lqqqqqqqqq^(-)qqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqk x
 x x         [ ] Export task/process statistics through netlink (EXPERIMENTAL)            x x
 x x         [*] Auditing support                                                         x x
 x x             RCU Subsystem  --->                                                      x x
 x x         <*> Kernel .config support                                                   x x
 x x         [*]   Enable access to .config through /proc/config.gz                       x x
 x x         (15) Kernel log buffer size (16 => 64KB, 17 => 128KB)                        x x
 x x         [ ] Control Group support  --->                                              x x
 x x         [ ] enable deprecated sysfs features to support old userspace tools          x x
 x x         [ ] Kernel->user space relay support (formerly relayfs)                      x x
 x x         [ ] Namespaces support                                                       x x
 x x         [*] Initial RAM filesystem and RAM disk (initramfs/initrd) support           x x
 x x         (/home/keng/local/project/highmem/linux-mti-2.6.35.9-2/test/out/hm) Initramfs sx x
 x x         (0)     User ID to map to 0 (user root) (NEW)                                x x
 x x         (0)     Group ID to map to 0 (group root) (NEW)                              x x
 x x         [*]   Support initial ramdisks compressed using gzip                         x x
 x mqqqqqqqqv(+)qqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqj x
 tqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqu
 x                             <Select>    < Exit >    < Help >                             x
 mqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqj
```

When completed, save the configuration file and continue to the next step.

2.3 Limit Memory Usable to 128 MBytes

By default, the function void __init prom_meminit(void) in ./arch/mips/mti-malta/malta-memory.c on line 135 uses all memory available (in accordance with information passed by the boot loader). To limit available memory to 128 MB, modify the function void_init prom_meminit(void) as shown below.

```
void __init prom_meminit(void)
{
        struct prom_pmemblock *p;

#ifdef DEBUG
        pr_debug("YAMON MEMORY DESCRIPTOR dump:\n");
        p = prom_getmdesc();
        while (p->size) {
                int i = 0;
                pr_debug("[%d,%p]: base<%08lx> size<%08lx> type<%s>\n",
                         i, p, p->base, p->size, mtypes[p->type]);
                p++;
                i++;
        }
#endif
        p = prom_getmdesc();

        while (p->size) {
                long type;
                unsigned long base, size;

                type = prom_memtype_classify(p->type);
                base = p->base;
                size = p->size;

                if (type==1) {
                        p->size=134217728- p->base;
                        size= p->size;
                }

                add_memory_region(base, size, type);
                p++;
        }
}
```

We are limiting reserved plus free memory to 128 MB. The size is calculated from the following:

0x08000000-000f0000-00aae000 = 0x7462000 (122,036,224)

After making the above change, the following memory map will appear on the Linux boot message:

```
Determined physical RAM map:
 memory: 00001000 @ 00000000 (reserved)
 memory: 000ef000 @ 00001000 (ROM data)
 memory: 00aae000 @ 000f0000 (reserved)
 memory: 07462000 @ 00b9e000 (usable)
```

2.4 Register Memory Used as HighMem

For HighMem support, the kernel must be hardcoded so that it knows there is an additional memory region available to the system, and that Linux must register it for HighMem. To do this, two files must be changed.

First, modify the function void __init prom_init(void) in arch/mips/mti-malta/malta-init.c , line 360:

```
--- linux-mti-2.6.35.9-2/arch/mips/mti-malta/malta-init.c       2011-09-20 14:43:09.000000000 -0700
+++ all/arch/mips/mti-malta/malta-init.c        2012-06-26 17:17:36.831485179 -0700
@@ -357,6 +357,9 @@

        prom_init_cmdline();
        prom_meminit();
+printk(KERN_INFO "==================register 64 MB memory for HighMem===================\n");
+add_memory_region(0x08800000, 0x04000000, BOOT_MEM_RAM);
+
 #ifdef CONFIG_SERIAL_8250_CONSOLE
        console_config();
 #endif
```

In this example, 64 MB (0x04000000) of memory is added at address PA=0x08800000. The address 0x08800000 is used instead of 0x08000000 (which is immediately after the 128 MB RAM that uses normal memory), because otherwise the kernel would merge 64 MB with 128 MB, and no HighMem would be available.

Second , modify static void __init bootmem_init(void) in arch/mips/kernel/setup.c, line 316:

```
--- linux-mti-2.6.35.9-2/arch/mips/kernel/setup.c       2011-09-15 16:05:03.000000000 -0700
+++ all/arch/mips/kernel/setup.c        2012-06-25 13:10:12.174673559 -0700
@@ -313,13 +313,19 @@
         * Determine low and high memory ranges
         */
        max_pfn = max_low_pfn;
-       if (max_low_pfn > PFN_DOWN(HIGHMEM_START)) {
+       printk(KERN_INFO "keng==HIGHMEM_START==%ldk %ld \n",HIGHMEM_START, PFN_DOWN(HIGHMEM_START));
+       #undef HIGHMEM_START
+       #define HIGHMEM_START   128*1024*1024
+       //if (max_low_pfn > PFN_DOWN(HIGHMEM_START)) {
+       if (1) {
 #ifdef CONFIG_HIGHMEM
               highstart_pfn = PFN_DOWN(HIGHMEM_START);
               highend_pfn = max_low_pfn;
-#endif
               max_low_pfn = PFN_DOWN(HIGHMEM_START);
+#endif
+               printk(KERN_INFO "keng==HIGHMEM_START==%ld %ld %ld\n",highstart_pfn, highend_pfn,max_low_pfn);
        }
+       printk(KERN_INFO "keng==max_low_pfn > PFN_DOWN(HIGHMEM_START)==%ld %ld %ld\n",highstart_pfn,
highend_pfn,max_low_pfn);

        /*
        * Initialize the boot-time allocator with low memory only.
```

By default, the MIPS Linux kernel defines HIGHMEM_START to start after 512 MB. In this example, HIGHMEM_START is redefined to start after 128 MB.

3 Build and Run the kernel

Use the following command to build the kernel:

```
$ make CROSS_COMPILE=mips-linux-gnu-
  CHK     include/linux/version.h
  CHK     include/generated/utsrelease.h
  CC      scripts/mod/empty.o
  MKELF   scripts/mod/elfconfig.h
  HOSTCC  scripts/mod/file2alias.o
  HOSTCC  scripts/mod/modpost.o
  HOSTCC  scripts/mod/sumversion.o
  HOSTLD  scripts/mod/modpost
:
:
:
  CC      net/sched/sch_teql.mod.o
  LD [M]  net/sched/sch_teql.ko
  CC      net/xfrm/xfrm_ipcomp.mod.o
  LD [M]  net/xfrm/xfrm_ipcomp.ko
  CC      net/xfrm/xfrm_user.mod.o
  LD [M]  net/xfrm/xfrm_user.ko

$ mips-linux-gnu-objcopy -O srec vmlinux.srec
```

Copy the file vmlinux.srec to the tftp server, and on the YAMON console, do the following:

```
load tftp://192.168.11.1/highmem/vmlinux.srec
go . init=/init ip=dhcp
```

You should see a message similar to the kernel boot up message below. Message lines marked in red indicate that HighMem is registered correctly. A total of 192 MB should be available for the system.

```
Start = 0x80104150, range = (0x80100000,0x80b69f57), format = SREC
Linux version 2.6.35.9up (keng@linux-softcsd) (gcc version 4.5.2 (Sourcery CodeBench Lite 2011.03-110) ) #42 Thu Jun 28 10:39:39
PDT 2012
Config serial console: console=ttyS0,38400n8r
bootconsole [early0] enabled
CPU revision is: 00019555 (MIPS 34Kc)
Software DMA cache coherency
Determined physical RAM map:
 memory: 00001000 @ 00000000 (reserved)
 memory: 000ef000 @ 00001000 (ROM data)
 memory: 00aae000 @ 000f0000 (reserved)
 memory: 07462000 @ 00b9e000 (usable)
 memory: 04000000 @ 08800000 (usable)
Wasting 95168 bytes for tracking 2974 unused pages
keng==HIGHMEM_START==-536870912k 917504
keng==HIGHMEM_START==32768 51200 32768
keng==max_low_pfn > PFN_DOWN(HIGHMEM_START)==32768 51200 32768
Initrd not found or empty - disabling initrd
Zone PFN ranges:
  DMA      0x00000000 -> 0x00001000
  Normal   0x00001000 -> 0x00008000
  HighMem  0x00008000 -> 0x0000c800
Movable zone start PFN for each node
early_node_map[2] active PFN ranges
    0: 0x00000000 -> 0x00008000
    0: 0x00008800 -> 0x0000c800
Built 1 zonelists in Zone order, mobility grouping on.  Total pages: 48752
Kernel command line: init=/init ip=dhcp console=ttyS0,38400n8r
PID hash table entries: 512 (order: -1, 2048 bytes)
Dentry cache hash table entries: 16384 (order: 4, 65536 bytes)
Inode-cache hash table entries: 8192 (order: 3, 32768 bytes)
Primary instruction cache 64kB, 4-way, VIPT, linesize 32 bytes.
Primary data cache 64kB, 4-way, VIPT, cache aliases, linesize 32 bytes
Writing ErrCtl register=00000000
Readback ErrCtl register=00000000
Memory--: 182984k/119176k available (3466k kernel code, 1728k reserved, 815k data, 6380k init, 65536k highmem)
Hierarchical RCU implementation.
        RCU-based detection of stalled CPUs is disabled.
        Verbose stalled-CPUs detection is disabled.
:
:
:
Please press Enter to activate this console.
#
#
#
#
# free
              total       used       free     shared    buffers
  Mem:       190320       9020     181300          0          0
 Swap:            0          0          0
Total:       190320       9020     181300
```

3.1 Test the HighMem

The simplest way to test the correct functioning of HighMem is to write a user-space application that uses malloc to allocate memory greater than, in this case, 128 MB.

```
#include <stdlib.h>


//#define maxnum        200
#define maxsize 1024*1024

int main ()
```

```
{
  int i,n;

  unsigned char *buffer[maxnum];
  unsigned char *bufptr;

        for (i=0;i<maxnum;i++) {
                buffer[i] = (unsigned char*) malloc (maxsize);
                if (buffer[i]==NULL) {
                        printf("fail to allocate memory %d\n",i);
                        exit (1);
                }
        }

        // init buffer
        for (i=0;i<maxnum;i++) {
                bufptr=buffer[i];
                for (n=0;n<maxsize;n++) {
                        *bufptr=i;
                        bufptr++;
                }
        }

        // test buffer
        for (i=0;i<maxnum;i++) {
                bufptr=buffer[i];
                for (n=0;n<maxsize;n++) {
                        if (*bufptr!=i) {
                                printf("data not match c=%d i=%d, n=%d\n",*bufptr,i,n);
                                return 0;
                        }
                        bufptr++;
                }
        }

        // free buffer
        for (i=0;i<maxnum;i++) {
                free(buffer[i]);
        }

        printf("test complete\n");

  return 0;
}
```

To compile the code, use the following command:

```
$ mips-linux-gnu-gcc -msoft-float -EL -O3 -funroll-loops -static -o memtest240 -Dmaxnum=240 memtest.c
```

The -Dmaxnum flag is used to control the size of memory to allocate and test.

4 Q & A

4.1 What is the meaning of the message "Wasting 95168 bytes for tracking 2974 unused pages" shown in the Malta Linux boot message?

This message appears because the reserved memory and the ROM data memory map are registered with the kernel. The total size of these memories is 0x1000 + 0xef000 + 0xaae000 = 0xb9e000. With a 4 KB page size, the total number of pages is 0xb9e000 / 0x1000= 2,974. This number is calculated before HighMem is registered (in ./arch/mips/kernel/setup.c). Thus, it hasn't accounted for the page entry that is required for the "hole" between normal memory and HighMem. For each page entry, it's 32 bytes.

```
Determined physical RAM map:
 memory: 00001000 @ 00000000 (reserved)
 memory: 000ef000 @ 00001000 (ROM data)
 memory: 00aae000 @ 000f0000 (reserved)
 memory: 07462000 @ 00b9e000 (usable)
 memory: 04000000 @ 08800000 (usable)
Wasting 95168 bytes for tracking 2974 unused pages
```