

# 多機能マイコンボード Kane BeBe H8/3069F

組み込み Linux 超入門

マニュアル

株式会社日昇テクノロジー

<http://www.csun.co.jp>

[info@csun.co.jp](mailto:info@csun.co.jp)

2009/11/15



[copyright@2009](http://www.csun.co.jp)

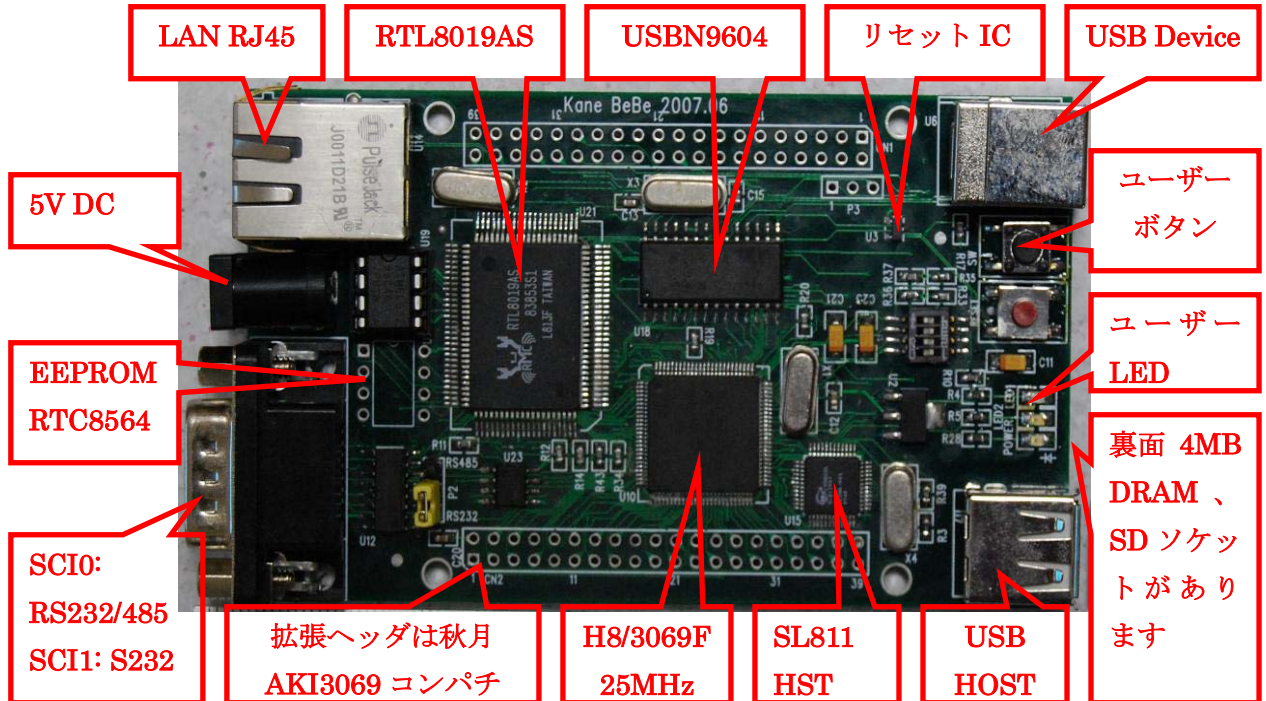


第一章 Kane BeBe H8/3069F ボードの概要.....	3
1.1 仕様.....	3
1.2 使えるデバイス例.....	5
第二章 Kane BeBe H8/3069 の操作方法.....	6
2.1 パソコンを繋ぐ.....	6
2.2 パソコン側のハイパーターミナルの設定.....	7
2.3 Kane BeBe H8/3069F の書き込み.....	11
第三章 TOPPERS JSP/TINET.....	12
3.1 お勧めの本.....	12
3.2 開発ツールのインストール.....	12
3.3 TOPPERS インストール手順.....	13
3.4 初心者の入門プログラム.....	13
3.5 TOPPERS/JSP を実行する.....	13
3.6 TOPPERS のウェブサーバー.....	15
第四章 uClinux.....	17
4.1 uClinux を選ぶ理由.....	17
4.2 お勧めの本.....	18
4.3 uClinux の初体験.....	18
4.4 SD カードがルートファイルとして uClinux を起動.....	24
4.5 NFS(Network File system)がルートファイルとして.....	25
第五章 uClinux の開発.....	26
5.1 開発環境のインストール.....	26
5.2 uClinux のコンフィグとコンパイル.....	27
5.3 uClinux-2.6.12 の NFS のコンフィグ例.....	35
5.3 uClinux 環境で自作プログラムを作る.....	42
5.4 アプリケーションを開発する場合の注意事項.....	44
5.5 液晶 NOKIA3310 を使用する.....	45

※ 使用されたソースコードは<http://www.csun.co.jp/>からダウンロードできます。

## 第一章 Kane BeBe H8/3069F ボードの概要

### 1.1 仕様



#### CPU プロセッサ

- ルネサステクノロジーの H8/3069F、周波数 25MHz。

#### メモリ

- 4MB FastPage DRAM メモリ
- 512KB 内蔵 Flash

#### インターフェース

- 10Base-T Ethernet RJ45(RTL8019AS) x 1
- シリアルポート SCI0: RS232/485, SCI1: RS232
- USB1.1 ホスト(SL811HST) x 1
- USB1.1 デバイス(USBN9604) x 1
- MMC/SD カードのソケット x 1

- ユーザーLED x 2
- ユーザーボタン x 1
- I2C バス x 1、AT24CXX と RTC8564 が使えます
- 拡張ヘッダのピン配列と寸法は秋月電子の AKI H8/3069F コンパチです。互換できます。

## 搭載した OS



uClinux

2.4.31/2.6.12




TOPPERS-1.4.3/TINET-1.3.2

- MES2.3r14(デフォルトの書き込み、MES から uClinux と TOPPERS がブートできます)

## 外形寸法

- 103×61(mm) 突起物は除く

## 供給電源

- 5VDC 電源、プラグ 2.1mmφ、極性はセンタープラス  です。電源指示 LED 付き

## メモリマップ

200000H	RTL8019AS
400000H	DRAM0
600000H	DRAM1
800000H	SL811HST
800004H	USBN9604

## 1.2 使えるデバイス例



外付けハードディスク



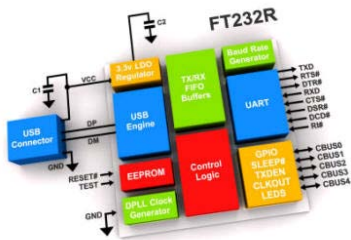
2GBまでの  
SD/MMC メモリ



USB HUB



USB メモリ



USB シリアルポート



USB マウスとキーボード



※ 付属のドライバ以外は、使えない可能性があります。

## 第二章 Kane BeBe H8/3069 の操作方法

### 2.1 パソコンを繋ぐ



Kane BeBe H8/3069F がパソコンを繋ぐのは DB9 メス-メス型のクロスケーブルを用意してください。



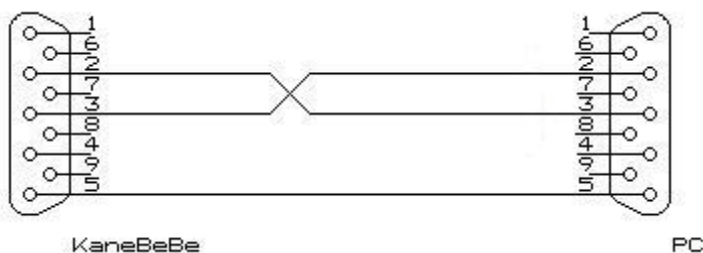
パソコンに RS232 ポートがなければ、USB-RS232 変換アダプタを使用してください。

※ RS232 クロスケーブルと USB-RS232 変換アダプタ別売

Kane BeBe H8/3069F が持っている二つシリアルポート(SCI0, SCI1)の信号は一つの DB9 から出るので、DB9 のピン配置は標準の RS232 ポートとちょっと違います。

DB9 ピン号	1	2	3	4	5	6	7	8	9
機能定義	TxD0	RxD1	TxD1	RxD0	GND	+5V	485A	485B	GND

パソコンと Kane BeBe を繋ぐ様子：



※ 市販の RS232 クロスケーブルも使えます。

## 2.2 パソコン側のハイパーターミナルの設定

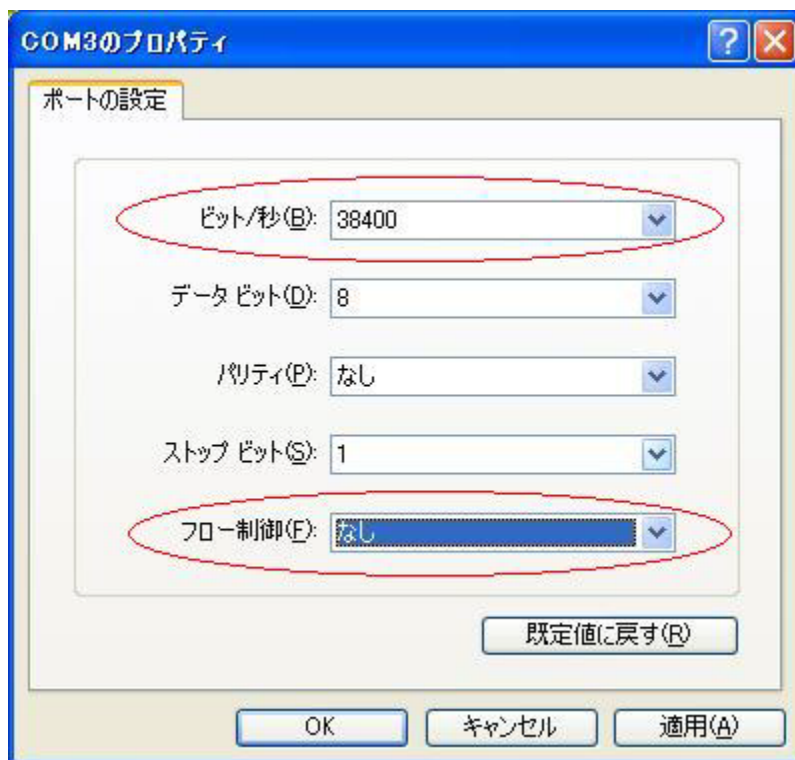
パソコンのメニュー：スタート → すべてのプログラム → アクセサリ → 通信 → ハイパーターミナルを選ぶと、次の画面が出てきます。



このハイパーターミナルの名前を入力して、"OK"ボタンを押すと。



使用したいシリアルポートを選んでください。



MES と TOPPERS と uClinux の間が互換できるために、シリアル通信速度を 38400bps に設定します。フロー制御はなしです。

"OK"ボタンを押すと、設定が完了します。

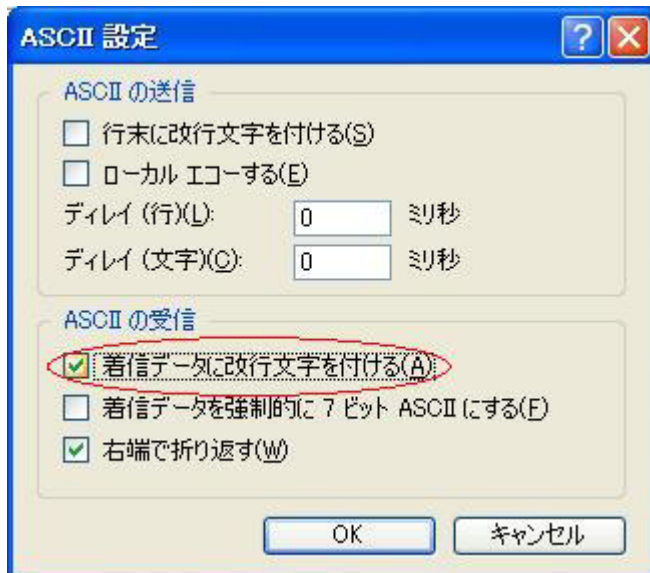
**MES** の場合は次の設定が必要です。

ハイパーターミナルのメニュー → ファイル → プロパティを選択して





プロパティの設定の[ASCII 設定]ボタンを押します。



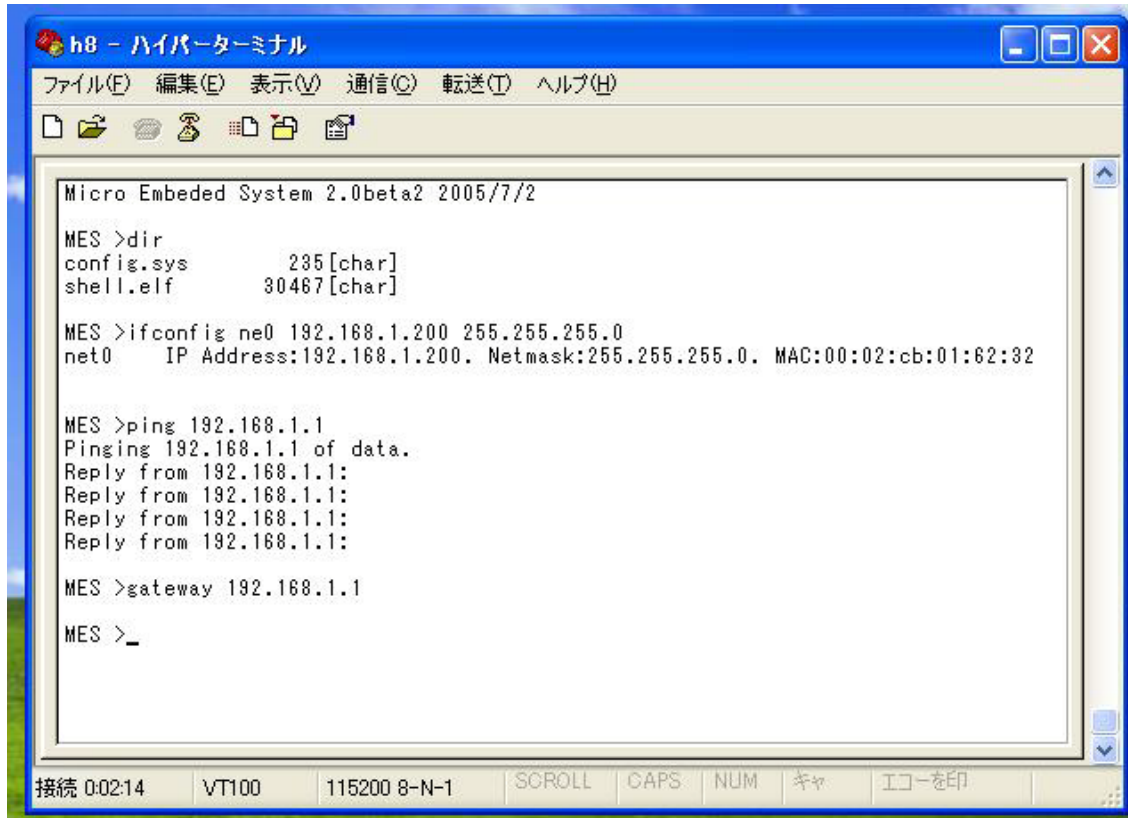
プロパティの設定の[ASCII 設定]ボタンをおします。

"着信データに改行文字を付ける"をチェックします。

MES の付加設定が完了しました。

ハイパーターミナルを設定完了したら、パソコンは Kane BeBe H8/3069F と通信できます。

下は通信の画面です



```
h8 - ハイパーターミナル
ファイル(F) 編集(E) 表示(V) 通信(C) 転送(T) ヘルプ(H)
Micro Embeded System 2.0beta2 2005/7/2
MES >dir
config.sys          235 [char]
shell.elf          30467 [char]
MES >ifconfig ne0 192.168.1.200 255.255.255.0
net0   IP Address:192.168.1.200. Netmask:255.255.255.0. MAC:00:02:cb:01:62:32
MES >ping 192.168.1.1
Pinging 192.168.1.1 of data.
Reply from 192.168.1.1:
Reply from 192.168.1.1:
Reply from 192.168.1.1:
Reply from 192.168.1.1:
MES >gateway 192.168.1.1
MES >_
接続 0:02:14  VT100  115200 8-N-1  SCROLL  CAPS  NUM  キャ  エコーを印
```

## 2.3 Kane BeBe H8/3069F の書き込み

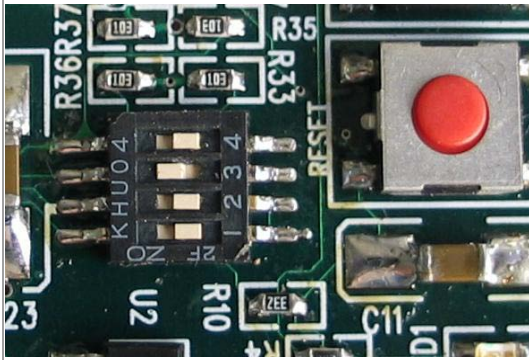
Kane BeBe H8/3069F には MES2.3 を書き込み済みでしたので、一般的に書き込みが必要ありません。DIP スイッチの状態は実行状態を設定します。

H8/3069F の内部の flash memory を更新すれば、書き込み状態を設定します。

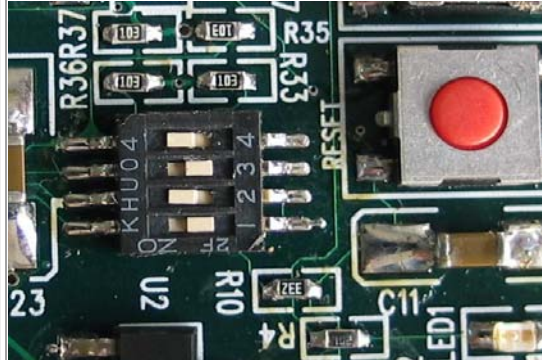
**！注意：状態変換の前には必ず電源を切ってください。**

### DIP スイッチの設定

#### 書き込み状態



#### 実行状態



1	2	3	4	1	2	3	4
off	off	on	off	off	on	Off	on

パソコン側の書き込みツールは三岩さんによって開発された h8flush.exe です。

h8flush.exe が動く様子：



## 第三章 TOPPERS JSP/TINET

### 3.1 お勧めの本



### 3.2 開発ツールのインストール

Cygwin 環境用開発ツール : h8300-hms-cygwin.tar.gz

Linux 環境用開発ツール : h8300-hms-linux-i686.tar.gz

```
# tar zxvf h8300-hms-cygwin.tar.gz -C /
```

または

```
# tar zxvf h8300-hms-linux-i686.tar.gz -C /
```



※ 使いやすいため、「**export PATH=\$PATH:/usr/local/h8300-hms/bin**」  
を.bashrc ファイルに入れてください。

## 3.3 TOPPERS インストール手順

1. Kene BeBe 用 TOPPERS/JSP + TINET(jsp-tinet.tar.gz)をダウンロードしてください。

2. Linux または Cygwin の中で解凍してください。

```
$ tar zxvf jsp-tinet.tar.gz
```

3. JSP のコンフィグ・ツールを生成する

```
cd jsp/cfg
```

```
make depend
```

```
make
```

4. TINET のコンフィグ・ツールを生成する

```
cd jsp/tinet/cfg
```

```
make
```

## 3.4 初心者の入門プログラム

```
$ cd jsp
```

```
$ mkdir APL
```

```
$ cd APL
```

```
$ perl ../configure -C h8 -S kbh8_3069f
```

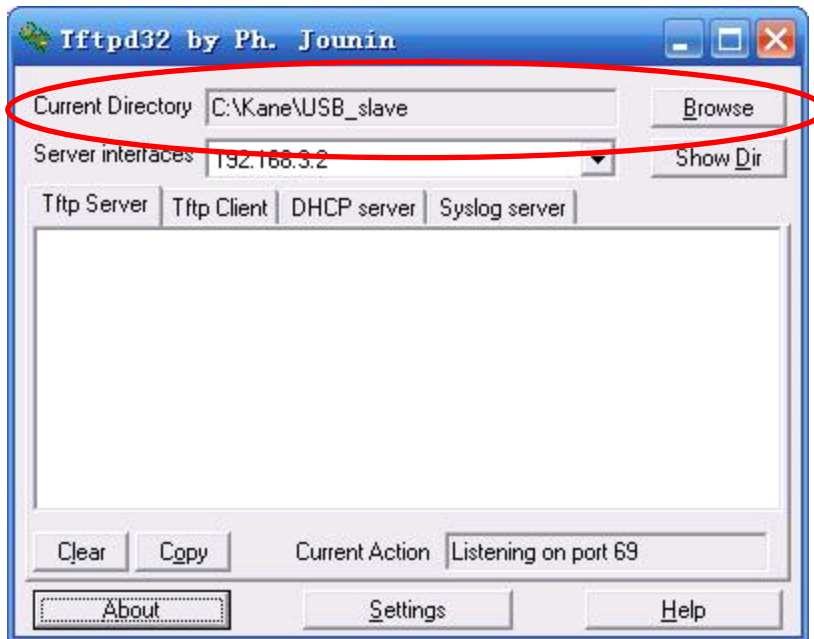
```
$ make depend
```

```
$ make
```

無事終了したら、TOPPERS の実行ファイル jsp.srec を生成します。

## 3.5 TOPPERS/JSP を実行する

1. パソコンでは tftp サーバー(tftpd32.exe)を実行してください。



生成された実行ファイル jsp.srec があるフォルダを選択してください。

Kane BeBe H8/3069F の環境中：

2. パソコンから TOPPERS のブートロードをダウンロードします。

**MES >tftp loadtop.elf 192.168.3.2**

※192.168.3.2 はパソコンの IP アドレスです。

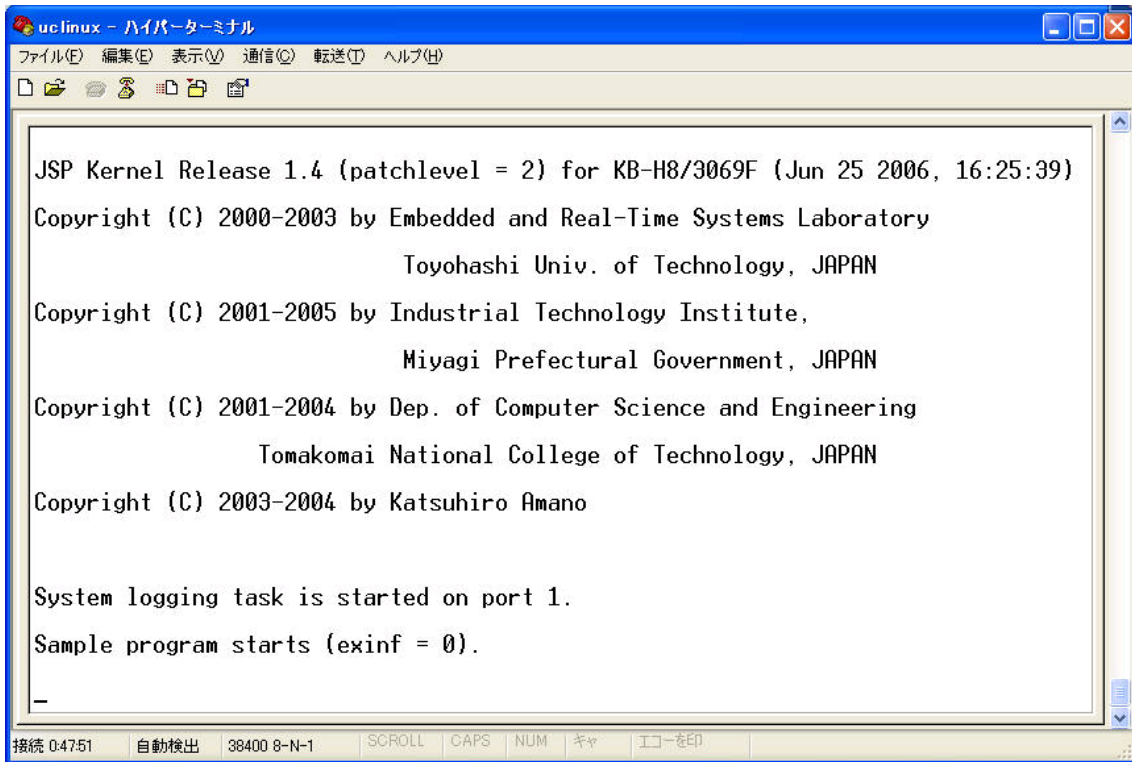
3. パソコンから TOPPERS の実行ファイルをダウンロードします

**MES >tftp jsp.srec 192.168.3.2**

4. TOPPERS を実行します。

**MES >loadtop.elf jsp.srec**

5. TOPPERS/JSP の起動画面



```
uclinux - ハイパーターミナル
ファイル(F) 編集(E) 表示(V) 通信(C) 転送(T) ヘルプ(H)
[Icons]
JSP Kernel Release 1.4 (patchlevel = 2) for KB-H8/3069F (Jun 25 2006, 16:25:39)
Copyright (C) 2000-2003 by Embedded and Real-Time Systems Laboratory
                        Toyohashi Univ. of Technology, JAPAN
Copyright (C) 2001-2005 by Industrial Technology Institute,
                        Miyagi Prefectural Government, JAPAN
Copyright (C) 2001-2004 by Dep. of Computer Science and Engineering
                        Tomakomai National College of Technology, JAPAN
Copyright (C) 2003-2004 by Katsuhiro Amano

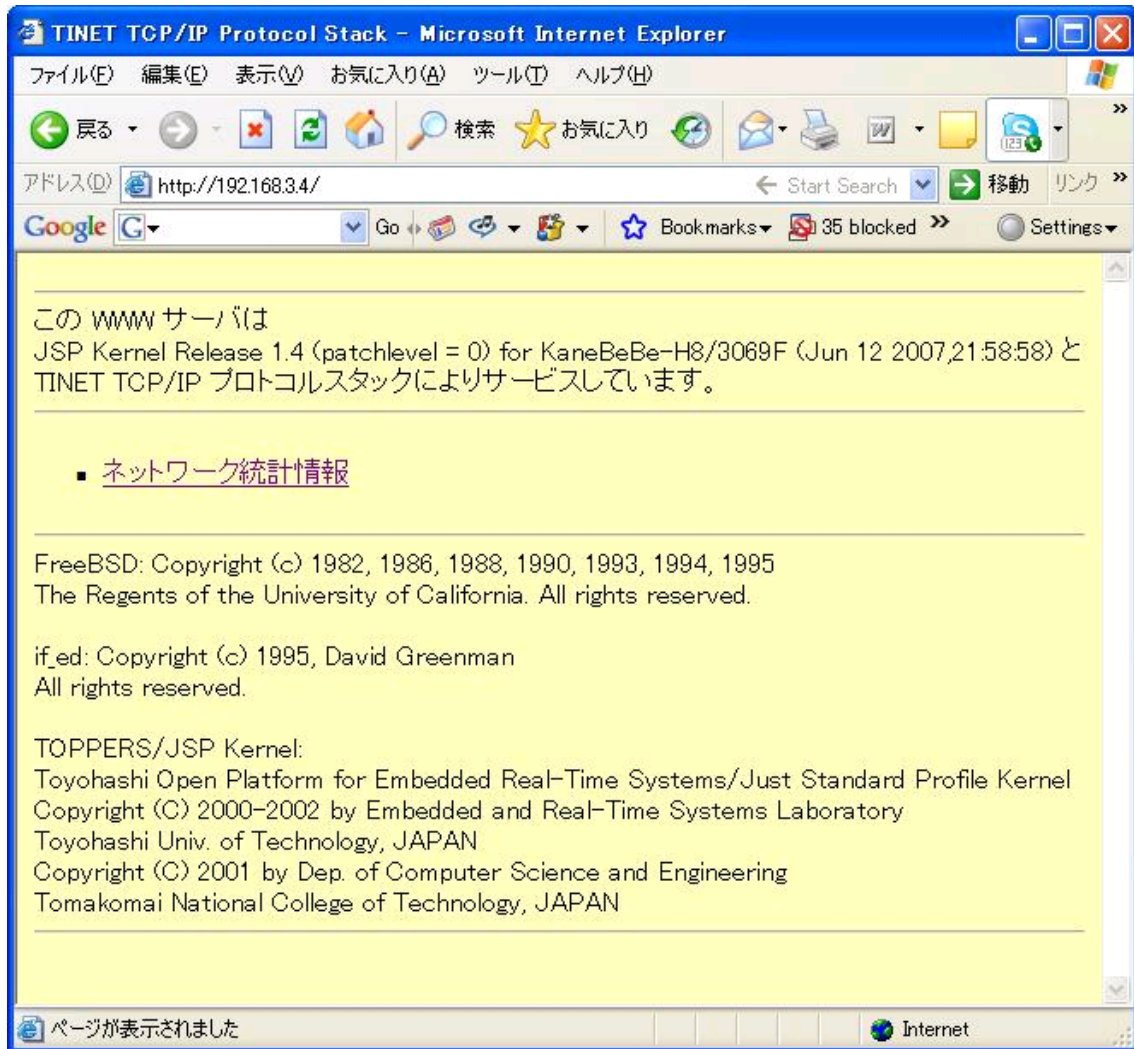
System logging task is started on port 1.
Sample program starts (exinf = 0).
_
接続 04751   自動検出   38400 8-N-1   SCROLL  CAPS  NUM  キャ  エコーを印
```

## 3.6 TOPPERS のウェブサーバー

```
$ cd jsp
$ cd nserv
$ make depend
$ make
```

無事終了したら、TOPPERS の実行ファイル `jsp.srec` を生成します。

`jsp.srec` を H8/3069F で実行したら、パソコンのブラウザで `http://192.168.3.4` を入力すると



※ TOPPERS のウェブサーバーの IP アドレスは `tinnet_app_config.h` の中で定義されています。



## 第四章 uClinux

### 4.1 uClinux を選ぶ理由



uClinux が動く様子(H8KANE と USB-HUB と CARD READER)

- **信頼性が高いネットワークスタックが利用できます。**  
uClinux には、Linux の TCP/IP スタックが実装され、利用できます。従って、ネットワークに接続する信頼性の高い機器が、容易に作成できる利点があります。
- **UNIX 系 OS の膨大なソフトウェア資産が使えます。**  
uClinux も UNIX 系 OS のシステムコールをサポートしています。従って、全てではありませんが、UNIX 系 OS 上のアプリケーションが組み込み用マイコン上で利用できます。ゼロから開発せず、例えば Web サーバなどが組み込み用機器で利用できるわけで、これは非常に大きな利点といえます。
- **デバッグが多少は楽になります。**  
UNIX 系 OS 上で動作するプログラムを組み込み用マイコンに実装できるため、UNIX 系 OS 上で動作を一通りテストすることができます。そのため、デバッグが多少は楽

になります。ただ、uClinuxはLinuxと異なる部分も少なくないため、パソコンで動くLinuxの上で、全てのテストができるわけではありません。

- **物理メモリアドレスに簡単にアクセスできます。**

uClinuxにはCPUの「特権モード」を使いません。仮想記憶がありませんから、アプリケーションがアクセスできるメモリ領域＝物理メモリアドレスです。そのため、uClinuxはユーザープログラムから任意の物理メモリアドレスに簡単にアクセスできます。さらに、割り込みすらユーザープログラムで受け取ることも可能です。

## 4.2 お勧めの本



初心者に対してとでもいい本です。原理、回路図、ソースコード、活用など詳しい説明しています。その上、サポートサイト <http://uclinux.quake4.jp> もあります。ご参照ください。

この本は秋月のAKI-H8/3069Fに基づいて、そのままKane BeBe H8/3069Fに流用することができます。

Kane BeBe H8/3069Fは4MBメモリがありますので、メモリ拡張が不要です、とても便利です。その上、Kane BeBe H8/3069FはUSB HOSTがありますので、uClinuxのUSB HOST機能も利用できます。

## 4.3 uClinuxの初体験

### 1. 生成されたuClinuxの体験版

uClinux-2.4.x: h8kane244.bin

uClinux-2.6.x: h8kane26.bin

MES環境のブートローダ:

loaduc.c(ソースファイル)loaduc.elf(実行ファイル)



2. SD メモリに loaduc.elf と h8kane26.bin または h8kane244.bin をコピーします。
3. H8/3069F に SD メモリカードを差し込み、H8/3069F の電源を入れます。

```
+-----+  
| Micro Embedded System Ver2.3 Rev14 |  
+-----+
```

MES >loaduc.elf h8kane26.bin console=ttySC1,38400n81

※ SD メモリカードには autoexec.bat ファイルがあれば、uClinux は自動的にブートローダできます。完璧に小さいな H8 マイコンボードがスタンドアロンで Linux マシンとして起動できます。すごいですね！

Now booting linux kernel:

Entry Address 0x00400000

Cmdline : console=ttySC1,38400n81

Linux version 2.6.12-uc0 (zqing@ip4.dragonwake.net) (gcc version 3.4.3)

#13 Thu

Jul 31 19:35:43 CST 2008

uClinux H8/300H

Target Hardware: H8KANE LAN/USB

Flat model support (C) 1998,1999 Kenneth Albanowski, D. Jeff Dionne

H8/300 series support by Yoshinori Sato

Built 1 zonelists

Kernel command line: console=ttySC1,38400n81

virtual vector at 0x00fffd20

PID hash table entries: 64 (order: 6, 1024 bytes)

Dentry cache hash table entries: 2048 (order: 1, 8192 bytes)

Inode-cache hash table entries: 1024 (order: 0, 4096 bytes)

Memory available: 1884k/610k RAM, 0k/0k ROM (1335k kernel code, 246k data)

Mount-cache hash table entries: 512

NET: Registered protocol family 16

SCSI subsystem initialized

usbcore: registered new driver usbfs

usbcore: registered new driver hub



```
SuperH SCI(F) driver initialized
ttySC0 at MMIO 0xfffffb0 (irq = 54) is a sci
ttySC1 at MMIO 0xfffffb8 (irq = 58) is a sci
io scheduler noop registered
ne-h8300.c:v1.00 2004/04/11 ysato
NE*000 ethercard probe at 00200000: 00 02 cb 01 62 32
eth0: NE1000 found at 0x200000, using IRQ 17.
uclinux[mtd]: RAM probe address=0x59dc08 size=0x6e000
Creating 1 MTD partitions on "RAM":
0x00000000-0x0006e000 : "ROMfs"
uclinux[mtd]: set ROMfs to be root filesystem
sl811: driver sl811-hcd, 19 May 2005
sl811-hcd sl811-hcd: SL811HS v1.5
sl811-hcd sl811-hcd: new USB bus registered, assigned bus number 1
sl811-hcd sl811-hcd: irq 12, io mem 0x00800000
hub 1-0:1.0: USB hub found
hub 1-0:1.0: 1 port detected
Initializing USB Mass Storage driver...
usbcore: registered new driver usb-storage
USB Mass Storage support registered.
h8mmc: H8/3069 SD/MMC card interface driver. Copyleft Qing Zhong
h8mmc CardType : SDSC
h8mmc: h8mmc1 h8mmc2 (MMC/SD メモリを挿入すれば、ある情報が出てきます。)
h8mmc : 124160Kbyte
NET: Registered protocol family 2
IP: routing cache hash table of 512 buckets, 4Kbytes
TCP established hash table entries: 512 (order: 0, 4096 bytes)
TCP bind hash table entries: 512 (order: -1, 2048 bytes)
TCP: Hash tables configured (established 512 bind 512)
VFS: Mounted root (romfs filesystem) readonly.
Freeing unused kernel memory: 48k freed (0x585000 - 0x590000)

Shell invoked to run file: /etc/rc
Command: hostname H8KANE
Command: mount -t proc proc /proc
Command: mount -t usbfs none /proc/bus/usb
```



Command: cat /etc/motd

Welcome to

```

      _ _ _
     /  _| ||_ |
    _  _| | | | _ _ _ _ _
   | | | | | | | | _ ¥ | | | | ¥ ¥ / /
   | |_| | |_| | | | | | |_| | /   ¥
   | ___ ¥ ___ | |_| |_| | |_| ¥ ___ | ¥ _/ ¥ _/
   | |
   |_|
```

CSUN/Kane BeBe H8/3069 (h8kane) port.

For further information check:

<http://www.uclinux.org/>

Execution Finished, Exiting

init: Failed to open /etc/inittab.

Sash command shell (version 1.1.1)

`> ifconfig eth0 192.168.3.4` (IP アドレスを設定します。)

`> ping 192.168.3.1`

PING 192.168.3.1 (192.168.3.1): 56 data bytes

64 bytes from 192.168.3.1: icmp\_seq=0 ttl=128 time=20.3 ms

64 bytes from 192.168.3.1: icmp\_seq=1 ttl=128 time=10.1 ms

64 bytes from 192.168.3.1: icmp\_seq=2 ttl=128 time=0.0 ms

64 bytes from 192.168.3.1: icmp\_seq=3 ttl=128 time=0.0 ms

--- 192.168.3.1 ping statistics ---

4 packets transmitted, 4 packets received, 0% packet loss

round-trip min/avg/max = 0.0/7.6/20.3 ms

`> thttpd -p 80 -d /var/www/ &` (ウェブサーバを起動します。)

`http://192.168.3.4` を訪問してみよう。)

[23]

`> free` (メモリの使用状態を見ます。)

MemTotal: 1976 kB

MemFree: 512 kB

Buffers: 152 kB



```
/> mount -t vfat /dev/h8mmc1 /mnt    (MMC/SD メモリの第 1 パーティション  
は FAT ファイルシステムです。)
```

```
/> ls /mnt
```

```
h8k242.bin    loaduc.c      h8kane26.bin  ext23.bin     ext2.bin  
vmlinux.bin  h8kane24.bin  ext23rom.bin  ext3.bin      loaduc.elf
```

```
/> umount /mnt
```

```
/> mount /dev/h8mmc2 /mnt          (MMC/SD メモリの第 2 パーティションは  
Linux のファイルシステムです。)
```

```
/> ls /mnt
```

```
var          tmp          proc         lib          etc          bin  
usr          sbin        mnt          home         dev          lost+found
```

```
/> umount /mnt
```

```
/> usb 1-1: new full speed USB device using sl811-hcd and address 2    (USB  
HUB を挿入すると、ある情報が出てきます。)
```

```
hub 1-1:1.0: USB hub found
```

```
hub 1-1:1.0: 4 ports detected
```

```
usb 1-1: new full speed USB device using sl811-hcd and address 12    (USB  
メモリを挿入すると、ある情報が出てきます。)
```

```
scsi3 : SCSI emulation for USB Mass Storage devices
```

```
Vendor: I-O DATA Model: USB Flash Disk Rev: 1.00
```

```
Type: Direct-Access ANSI SCSI revision: 02
```

```
SCSI device sda: 1024000 512-byte hdwr sectors (524 MB)
```

```
sda: Write Protect is off
```

```
sda: assuming drive cache: write through
```

```
ioctl_internal_command: <3 0 0 0> return code = 8000002
```

```
: Current: sense key=0x0
```

```
ASC=0x0 ASCQ=0x0
```

```
SCSI device sda: 1024000 512-byte hdwr sectors (524 MB)
```

```
sda: Write Protect is off
```

```
sda: assuming drive cache: write through
```

```
sda: sda1
```

```
Attached scsi removable disk sda at scsi3, channel 0, id 0, lun 0
```

```
/> mount -t vfat /dev/sda1 /mnt
```

```
ioctl_internal_command: <3 0 0 0> return code = 8000002
```



```
: Current: sense key=0x0
  ASC=0x0 ASCQ=0x0
/> ls /mnt
eCosDoc20.zip  thttpd
/> umount /mnt
/> cat /proc/version
Linux version 2.6.12-uc0 (zqing@ip4.dragonwake.net) (gcc version 3.4.3)
#1 Sat J
ul 19 12:39:14 CST 2008
/> cat /proc/gpio
P1: 00000000
P2: 00000000
P3: 11111111
P4: -----
P5: -----0
P6: 1----0--
P8: ----00-I
P9: --I-----
PA: -----
PB: 1000----
/> cat /proc/bus/usb/devices
T: Bus=01 Lev=00 Prnt=00 Port=00 Cnt=00 Dev#= 1 Spd=12 MxCh= 1
B: Alloc= 0/900 us ( 0%), #Int= 0, #Iso= 0
D: Ver= 1.10 Cls=09(hub ) Sub=00 Prot=00 MxPS= 8 #Cfgs= 1
P: Vendor=0000 ProdID=0000 Rev= 2.06
S: Manufacturer=uClinux 2.6.12-uc0 sl811-hcd
S: Product=SL811HS v1.5
S: SerialNumber=sl811-hcd
C:* #Ifs= 1 Cfg#= 1 Atr=c0 MxPwr= 0mA
I: If#= 0 Alt= 0 #EPs= 1 Cls=09(hub ) Sub=00 Prot=00 Driver=hub
E: Ad=81(I) Atr=03(Int.) MxPS= 2 IvL=255ms

T: Bus=01 Lev=01 Prnt=01 Port=00 Cnt=01 Dev#= 2 Spd=12 MxCh= 4
D: Ver= 1.10 Cls=09(hub ) Sub=00 Prot=00 MxPS= 8 #Cfgs= 1
P: Vendor=03eb ProdID=3301 Rev= 3.00
S: Product=Standard USB Hub
```

```
C:* #Ifs= 1 Cfg#= 1 Atr=e0 MxPwr= 64mA
I: If#= 0 Alt= 0 #EPs= 1 Cls=09(hub ) Sub=00 Prot=00 Driver=hub
E: Ad=81(I) Atr=03(Int.) MxPS= 1 IvI=255ms

T: Bus=01 Lev=02 Prnt=02 Port=02 Cnt=01 Dev#= 3 Spd=12 MxCh= 0
D: Ver= 2.00 Cls=00(>ifc ) Sub=00 Prot=00 MxPS=64 #Cfgs= 1
P: Vendor=04bb ProdID=0c06 Rev= 1.00
S: Product=USB Mass Storage Device
S: SerialNumber=02A000000001A2
C:* #Ifs= 1 Cfg#= 1 Atr=80 MxPwr=120mA
I: If#= 0 Alt= 0 #EPs= 3 Cls=08(stor.) Sub=06 Prot=50 Driver=usb-storage
E: Ad=01(0) Atr=02(Bulk) MxPS= 64 IvI=0ms
E: Ad=82(I) Atr=02(Bulk) MxPS= 64 IvI=0ms
E: Ad=83(I) Atr=03(Int.) MxPS= 64 IvI=8ms
/>
```

※ 赤い文字は入力されたコマンドまたはコメントです。

※ uClinux-2.4.x の場合は、MMC/SD と USB 機能がありません。

## 4.4 SD カードがルートファイルとして uClinux を起動

4.3 節の uClinux の初体験は H8/3069F のメモリの一部を Linux のルートファイルシステムとして起動します。簡単ですが、H8/3069F がある 4MB メモリも少ないので、できるだけ節約したほうがいいです。Linux の場合は、メモリが多ければ多いほどいいです。uClinux は SD カードをルートファイルシステムとして起動すれば、メモリが節約できます。

### 1. MMC/SD メモリのパーティション設定

Linux 環境で、fdisk コマンドを使い、SD のパーティションを作成してください。

```
# fdisk /dev/sdb
```

※ 本稿では MMC/SD メモリを USB で接続して /dev/sdb と認識されているという前提で話を進めています。以降は、読者の環境に応じてデバイスノード名を読み替えてください。

メモリカードを 2 つのパーティションに分けます。

### 2. フォーマット

パーティションを作成したら、フォーマットします。





```
# mkfs -t vfat /dev/sdb1    FAT ファイルシステムをフォーマットします。
```

```
# mkfs /dev/sdb2          uClinux のファイルシステムをフォーマットします。
```

※ MES は Linux で作成された FAT ファイルシステムを認識することができませんので、Windows で FAT パーティションを再フォーマットすることが必要です。

### 3. カーネルファイルのインストール

Linux カーネル: vmlinux.bin

MES 環境のブートローダ: loaduc.c(ソースファイル)loaduc.elf(実行ファイル)

vmlinux.bin と loaduc.elf を MMC/SD メモリカードの FAT パーティションにコピーします。

```
# mount -t vfat /dev/sdb1 /mnt
```

```
# cp vmlinux.bin /mnt
```

```
# cp loaduc.elf /mnt
```

```
# vi /mnt/autoexec.bat      (必要なら、autoexec.bat を編集します。)
```

```
# umount /mnt
```

### 4. ファイルシステムのインストール

ルートファイルシステム: rootfs.tar.gz

/dev/sdb2 がルートファイルシステムになります。

```
# mount /dev/sdb2 /mnt
```

```
# tar xvzf rootfs.tar.gz -C /mnt
```

```
# umount /mnt
```

```
# /sbin/modprobe -r usb-storage      (この後カードを抜き取る)
```

### 5. MES による uClinux-H8 起動

Kane BeBe H8/3069 ボードに MMC/SD メモリカードを差し込み、ボードの電源を入れます。(または、リセット)

MES が起動したら、以下の操作をします。

```
MES > loaduc.elf vmlinux.bin console=ttySC1,38400n81 root=/dev/h8mmc2
```

※ このコマンドを autoexec.bat に編集すれば、MMC/SD カードから uClinux が自動的に起動できます！

## 4.5 NFS(Network File system)がルートファイルとして

### 1. NFS サーバの設定

```
# mkdir /tftpboot/nfsroot
```



```
# tar xvzf rootfs.tar.gz -C /tftpbboot/nfsroot
# vi /etc/exports          (/etc/exports をエディタで開きます。)
「/tftpbboot/nfsroot *(rw, sync, no_root_squash)」 という行を/etc/exports に入れてくだ
さい。
# /etc/init.d/nfs restart  (NFS サーバを(再)起動します。)
```

## 2. MES による uClinux-NFS 起動

```
MES > loaduc.elf h8nfs26.bin console=ttySC1,38400n81 root=/dev/nfs
nfsroot=192.168.3.2:/tftpbboot/nfsroot
ip=192.168.3.10:192.168.3.2:192.168.3.1:255.255.255.0
```

## 3. RedBoot による uClinux-NFS 起動

```
RedBoot>load -r -v -b 0x400000 h8nfs26.bin -h 192.168.3.2
Using default protocol (TFTP)
|
Raw file loaded 0x00400000-0x005fb0c3, assumed entry at 0x00400000
RedBoot>exec -c "console=ttySC1,38400n81 root=/dev/nfs
nfsroot=192.168.3.2:/tftpbboot/nfsroot
ip=192.168.3.10:192.168.3.2:192.168.3.1:255.255.255.0"
```

- "ip="以降の IP アドレスは：  
192.168.3.10: H8KANE の IP アドレス  
192.168.3.2: bootserver の IP アドレス  
192.168.3.1: gateway の IP アドレス  
255.255.255.0: mask

※ firewall の設定によって、NFS を利用できない可能性があります。

## 第五章 uClinux の開発

### 5.1 開発環境のインストール

H8 のクロス開発環境 h8tools\_bin.tar.gz

uClinux-H8 カーネルソース

uClinux-dist-h8kane20081001.tar.gz(Nokia3310 のドライバを含む)



8\*8 ドットの漢字フォント・データをもたせているので、漢字ディスプレイとして使用できます。

門真なむさんの美咲フォントを利用しています。 <http://www.geocities.jp/littlimi/font.htm>

## 1. 開発ツールの解凍、root の権限が必要です。

```
# mkdir /h8tools
```

```
# tar zxvf h8tools_bin.tar.gz -C /h8tools
```

※ 使いやすいため、「**export PATH= \$PATH:/h8tools/bin**」を.bashrc  
ファイルに入れてください。

## 2. uClinux のソースファイルの解凍

```
$ tar zxvf uClinux-dist-h8kane20081001.tar.gz
```

## 5.2 uClinux のコンフィグとコンパイル

カレントディレクトリを uClinux-dist に移してください。

```
$ cd uClinux-dist
```

作業に入る前に

```
$ make clean
```

又は、

```
$ make dist-clean
```

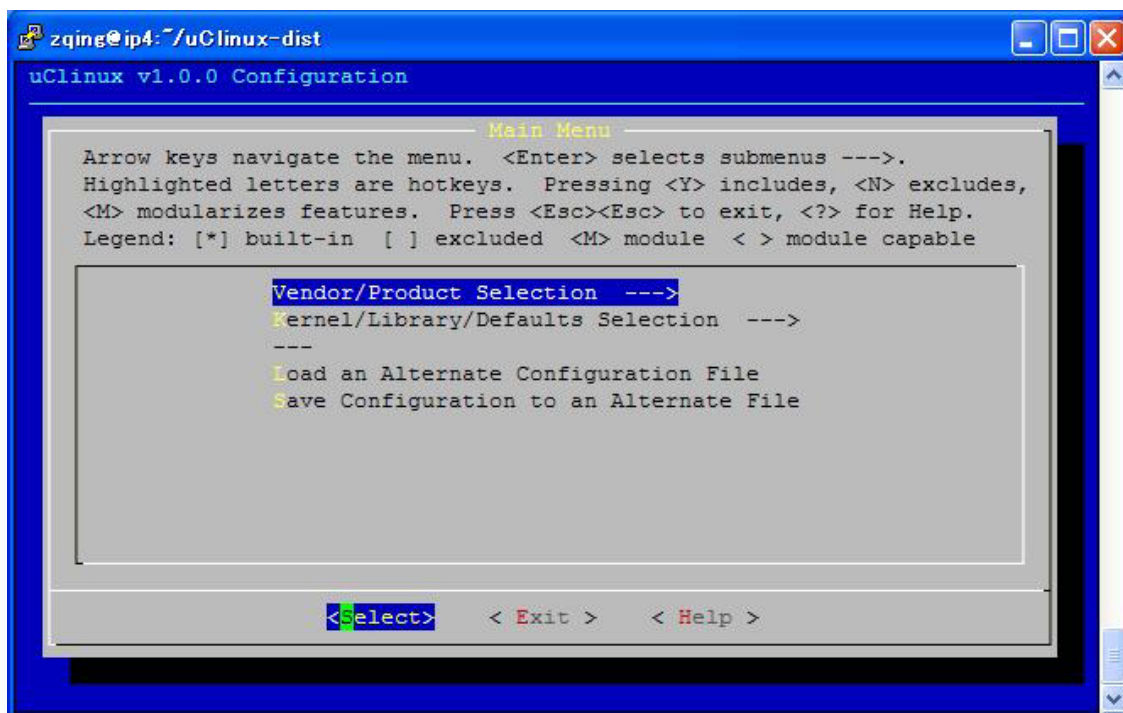
とコマンドを実行し、初期化しておいてください。

### 1. ターゲットの設定

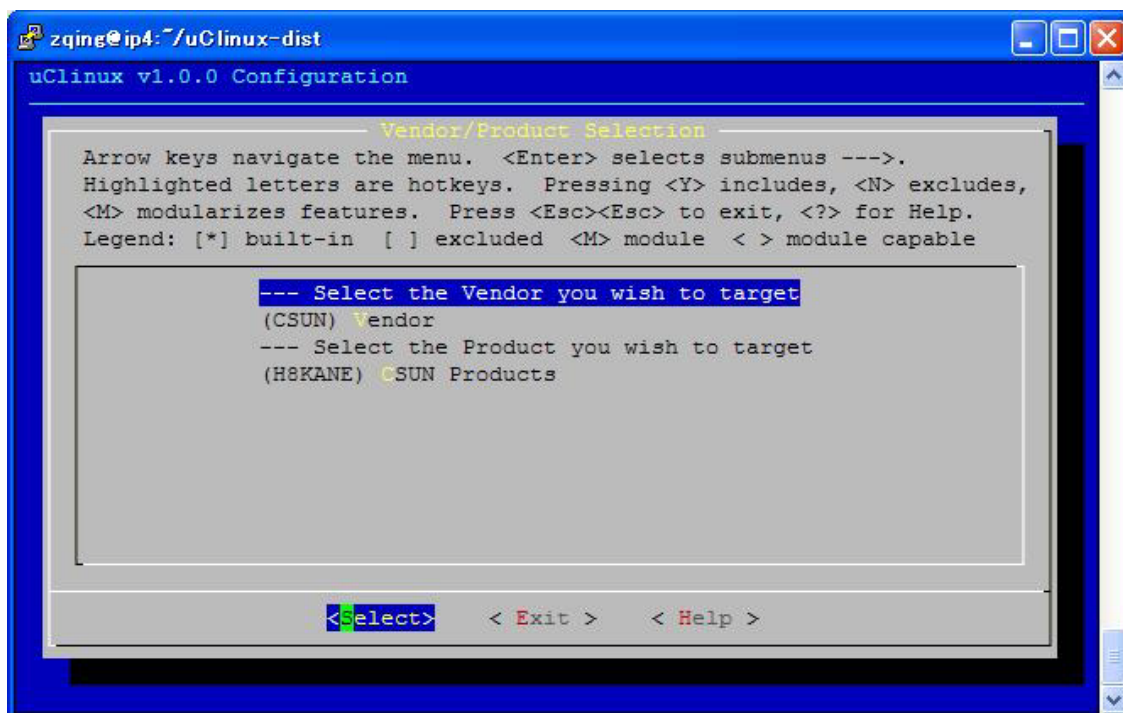
次のコマンドで設定メニューを起動します。

```
$make menuconfig
```

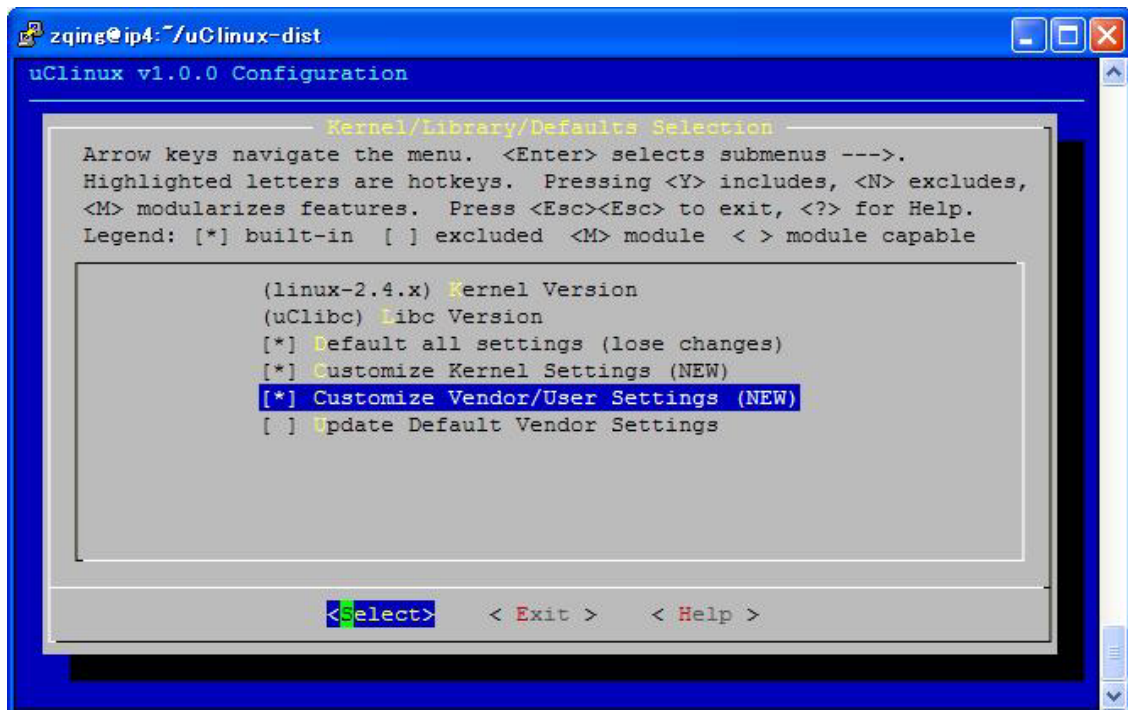
すると下の画面が開きます。



まず、「Vendor/Product Selection」を選択してください。ここで、uClinux を動作させるプラットフォームのベンダーおよび、製品名を設定します。Vendorを「CSUN」に、Productを「H8KANE」に設定します。



画面例のように設定したら、トップメニューに戻り、「Kernel/Library/Defaults Selection」を選択してください。

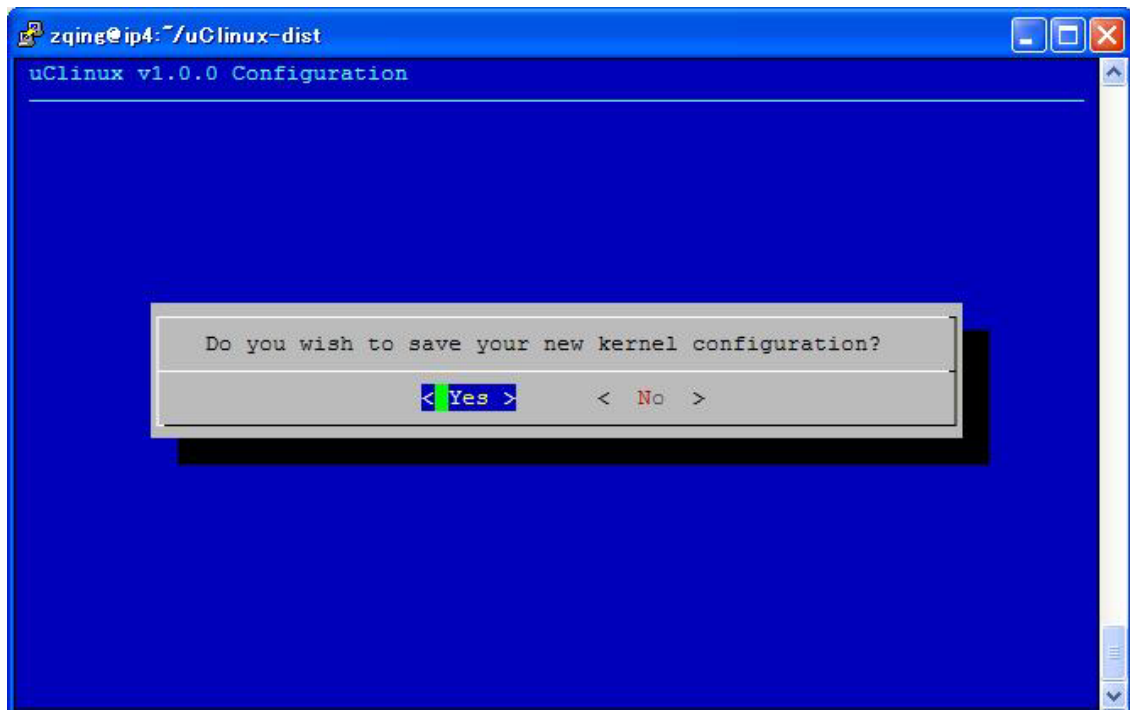


カーネルバージョンは 2.4.x を利用します。2.6.x の場合は 4MB メモリが必須です。「Default all settings」を選択すると、uClinux-dist に同梱されているデフォルトの設定が一度、適用されます。初回のビルド時にはチェックを入れてください(2 回目以降のビルド時には必要ありません)。

そして、「Customize Kernel Settings」にチェックを入れ、NFS ルートのための設定が行えるようにします。2MB メモリが少ないから、ROM ファイルシステムを使って uClinux を起動させることができますが、起動するだけでは何もできない、とっていいくらいの状態になってしまいます。NFS は UNIX 系 OS で標準的に利用されているディスク共有です。Linux は、NFS を利用してネットワーク経由でほかのコンピュータ上のファイルシステムをルートとしてマウントし起動する、「NFS ルート」をサポートしています。NFS ルートは、H8 マイコンのように自身にルートファイルシステムとして使えるようなブロックデバイスを持たない(いわゆるディスクレス)システムで、便利に利用できます。

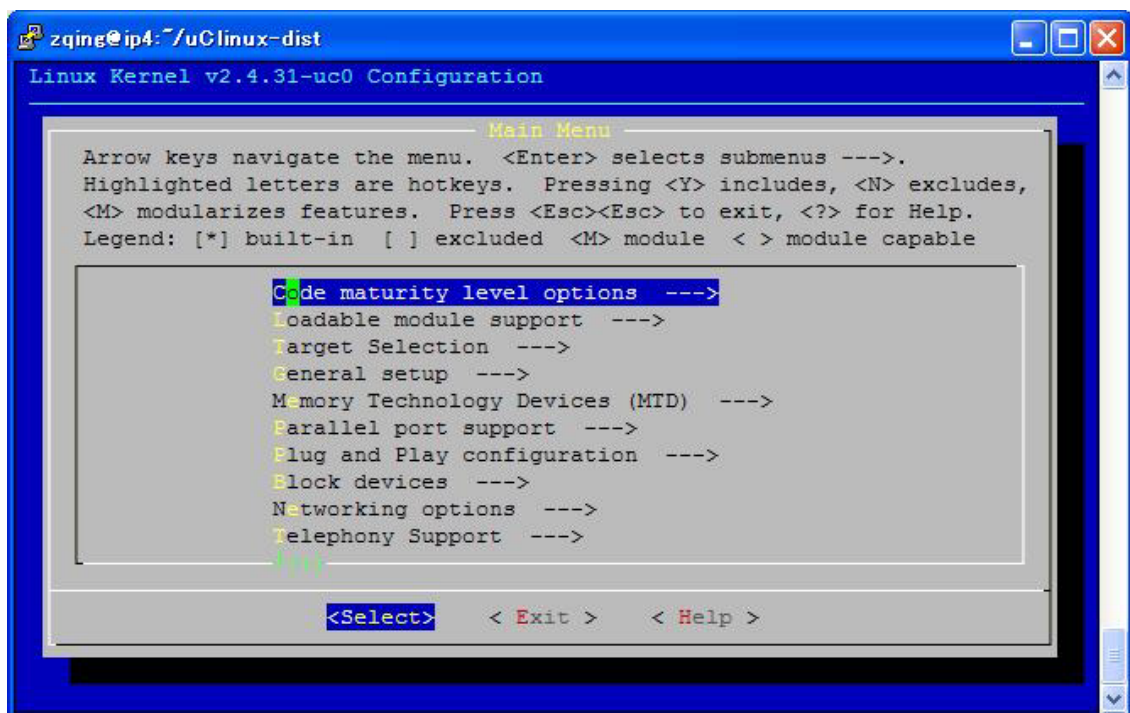
また、「Customize Kernel Settings」にチェックを入れておきましょう。Vendor/User settings は、uClinux で利用できるようにするコマンド類の設定です。

以上を設定したら、Exit を選択し、さらにトップメニューでも Exit を選択して設定の保存を指定して、メニューから抜けましょう。



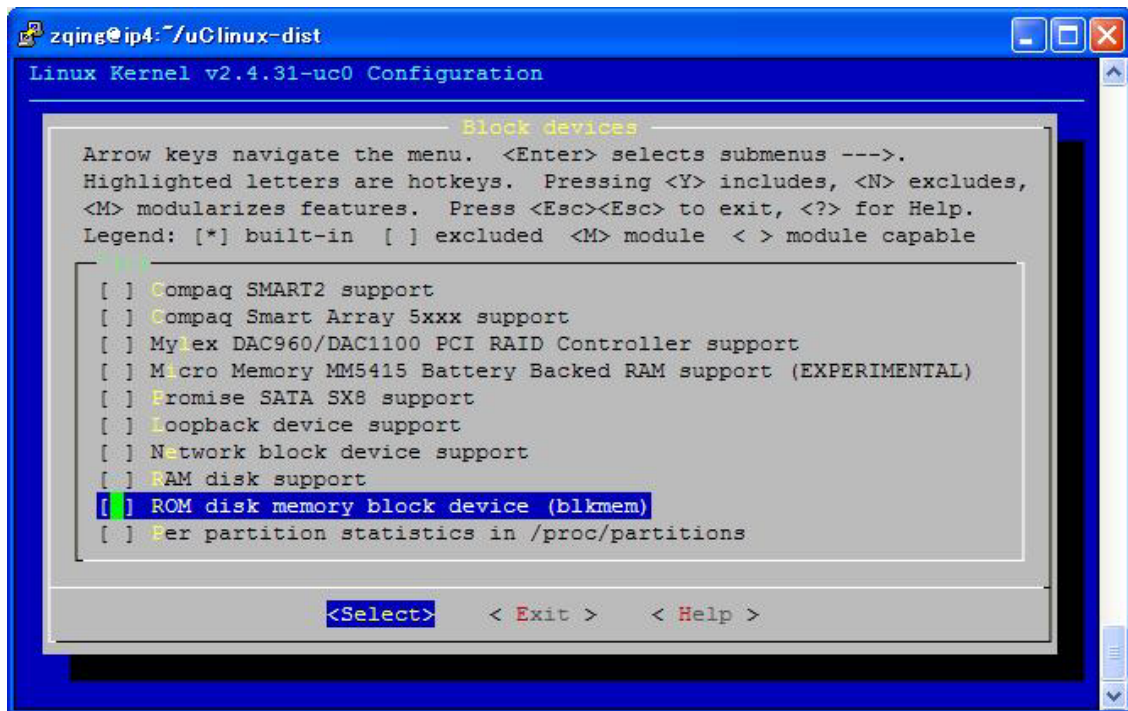
## 2. カーネル設定

ターゲットのメニューから抜けるとカーネルのデフォルト設定が適用され、自動的にカーネル設定メニューが起動してきます。

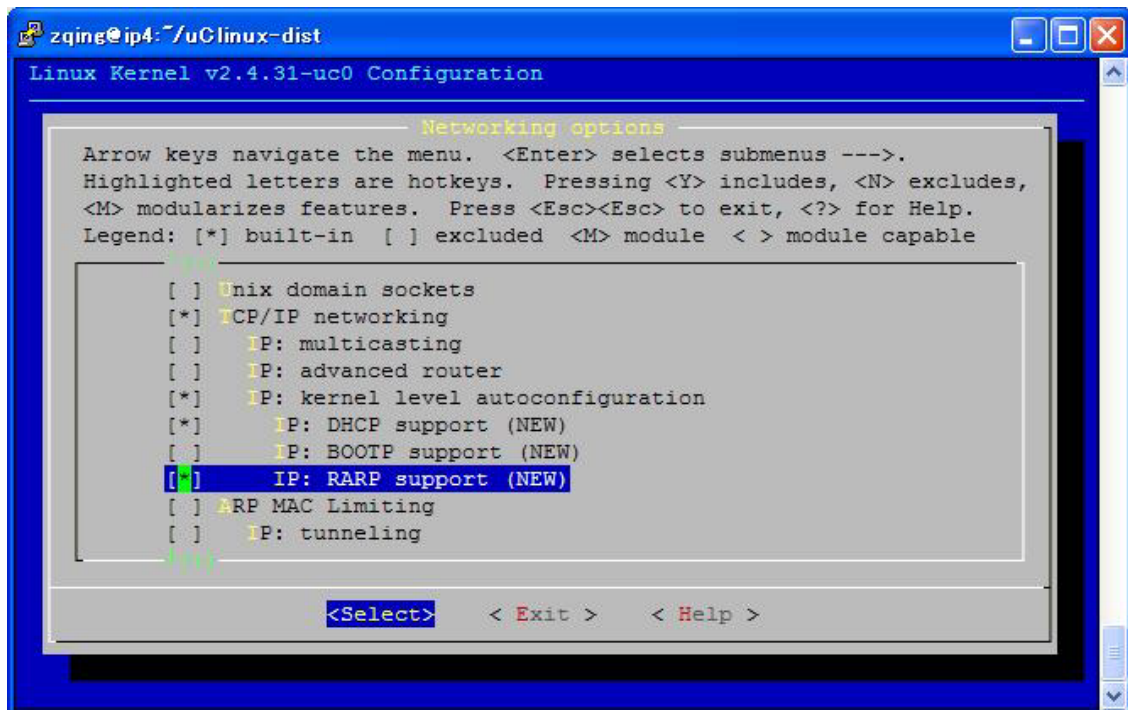


まず、「Block devices」を選択してください。メニューの末尾に「ROM disk memory block

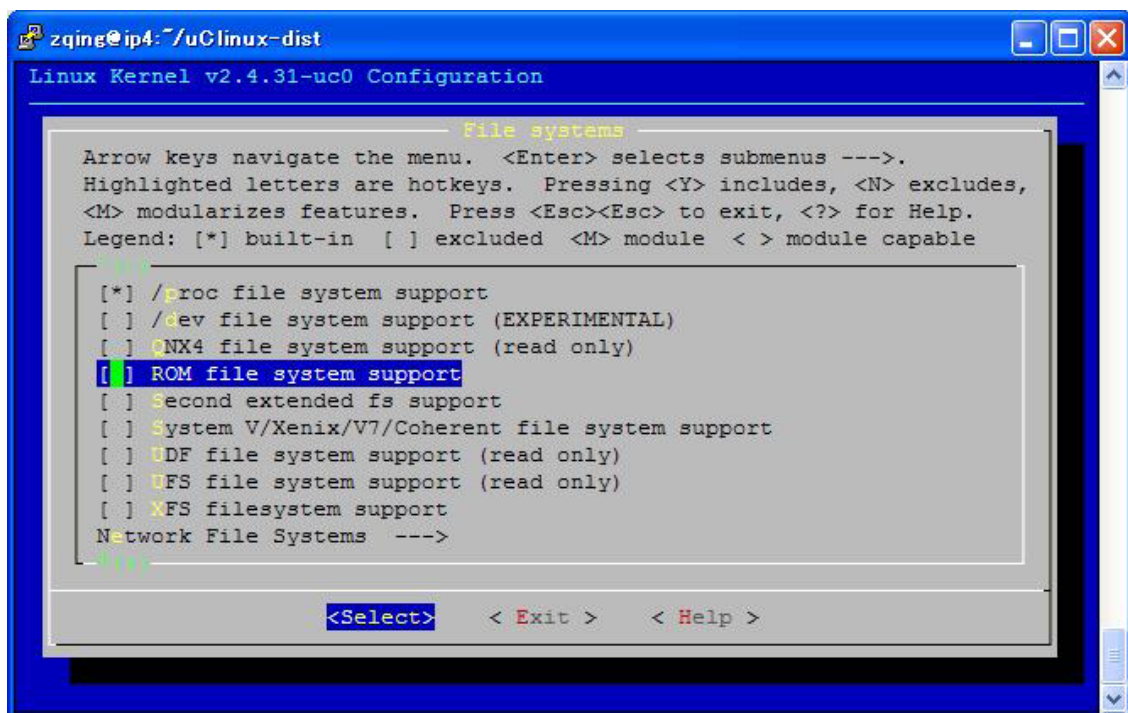
device(blkmem)」という項目があります。これが ROM ディスクサポートですので、スペースキーを押してオフ(無効)にします。



Exit を選択してカーネル設定のメニューに戻り、「Networking options」を選択します。設定項目の中から「kernel level autoconfiguration」を探してチェックを入れます。するとサブメニューが現れるので、「DHCP support」にもチェックを入れておいてください。これらの設定で、カーネル自身が DHCP サーバを参照して起動時に IP アドレスの自動設定が行われます。NFS をルートにマウントするためには、ルートをマウントする前に IP アドレスが設定されている必要があるため、カーネルに DHCP による自動設定を設定しておくわけです。



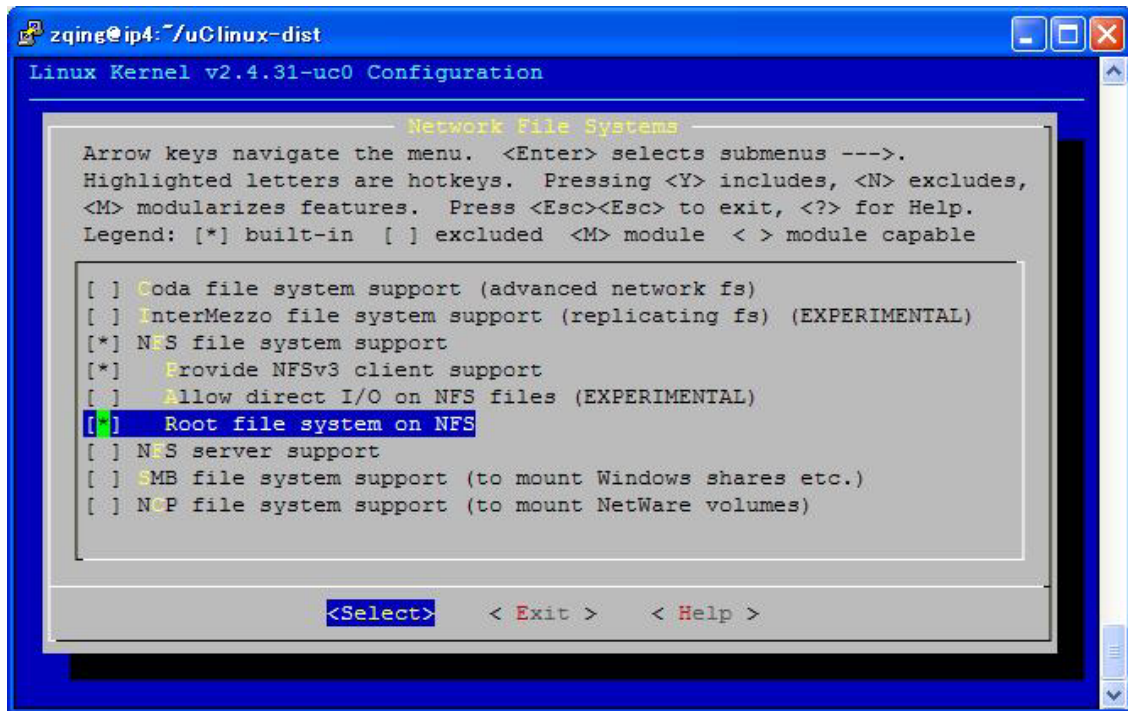
Networking options を例のように設定したら、Exit を選択してトップメニューに戻ります。File systems では、「Rom file system support」にチェックが入っていると思いますので、これをオフにします。



そして、File system メニューの下のほうにある「Network file Systems」という項目を選



お選びください。



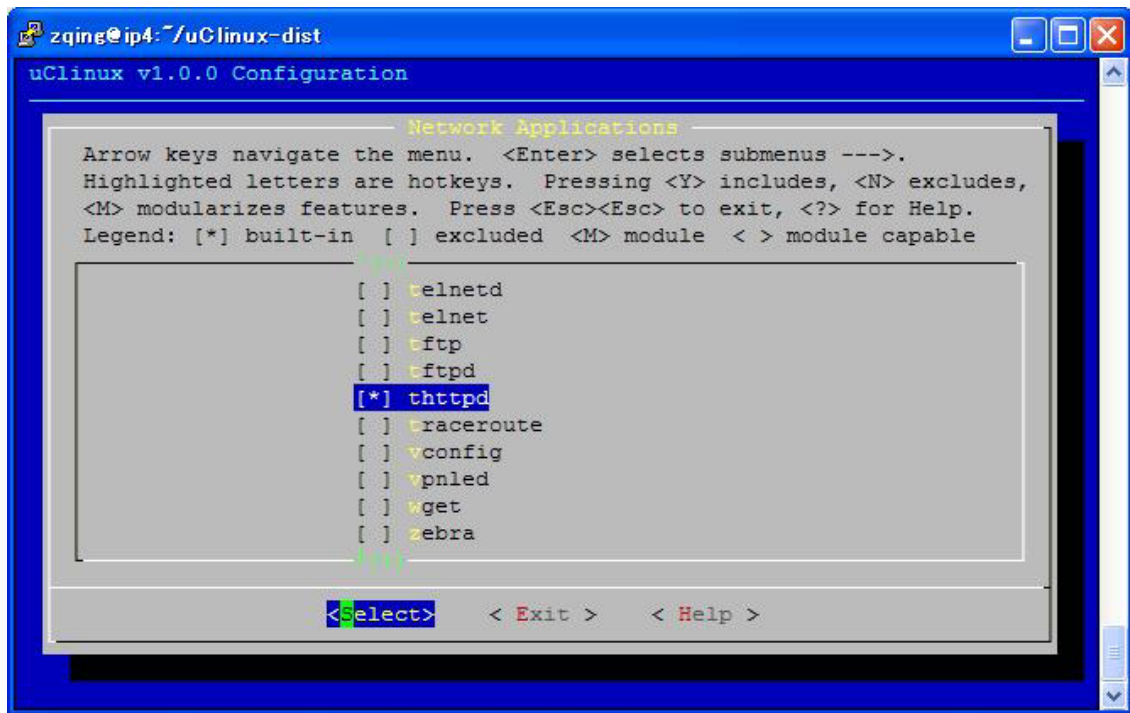
画面例のように「NFS file system support」をオンにし、「Provide NFSv3 client support」と「Root filesystem on NFS」(NFS ルート)をオンにします。

以上で NFS ルートのカーネルの設定は完了です。Exit を選択してメニューに戻り、さらに Exit を選択してメニューを終わらせてください。終了前に設定を保存するか聞いてくるので、Yes を選択します。

### 3. ユーザーランドの設定

カーネルの設定が保存されると、続いて自動的に設定のメニューが起動してきます。この設定では、主として作成する uClinux に導入するコマンドと、コマンドが利用するライブラリを設定します。ユーザーが操作するコマンド群ということで「ユーザーランド」などとも呼びます。

「Network Applications」には、Web サーバなどネットワークアプリケーションが分類されています。



今回はテストのためにシンプルな Web サーバである `t h t t p d` をビルドしておくことにします。スペースキーを押して選択状態にしておいてください。

以上のような設定を行ったら、`Exit` を選択してメニューを終了させ、設定を保存してください。

#### 4. ビルド

設定が保存できたら、次のコマンドを入力してビルドします。

```
$ make dep (linux-2.6.x の場合は、このコマンドが必要ありません。)
```

```
$ make
```

ビルドにはしばらく時間がかかります。コンパイル完了すれば、実行ファイルを生成します。

`linux-2.4.x/linux` (linux のカーネル、ELF フォーマット)

`romfs/` (uClinux のルートファイルシステム)

`images/h8kane-image.bin` (linux カーネル + romfs ルートファイルシステムのイメージファイル)

#### 5. 新 uClinux の実行

H8/3069F のコンソールで

MES > loaduc.elf linux .....パラメータ略

又は

MES > loaduc.elf h8kane-image.bin .....パラメータ略

起動のパラメータは前節を参照してください。

## 5.3 uClinux-2.6.12 の NFS のコンフィグ例

```
$ cd uClinux-dist
```

作業に入る前に

```
$ make clean
```

又は、

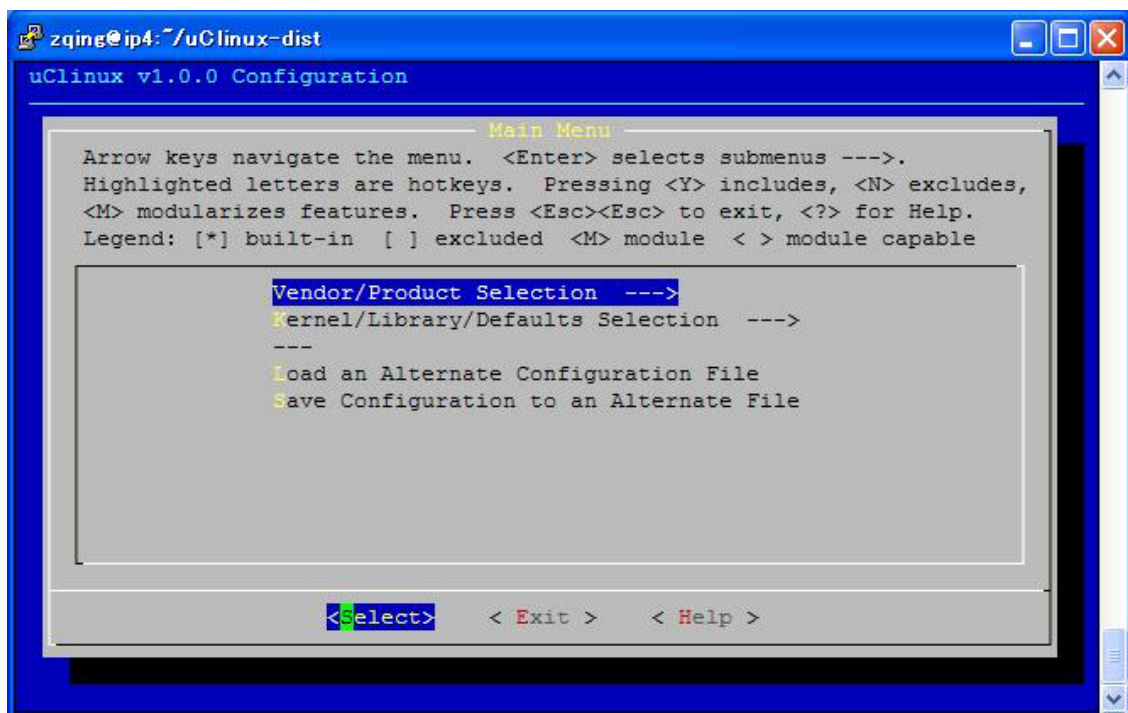
```
$ make dist-clean
```

とコマンドを実行し、初期化しておいてください。

### 1. ターゲットの設定

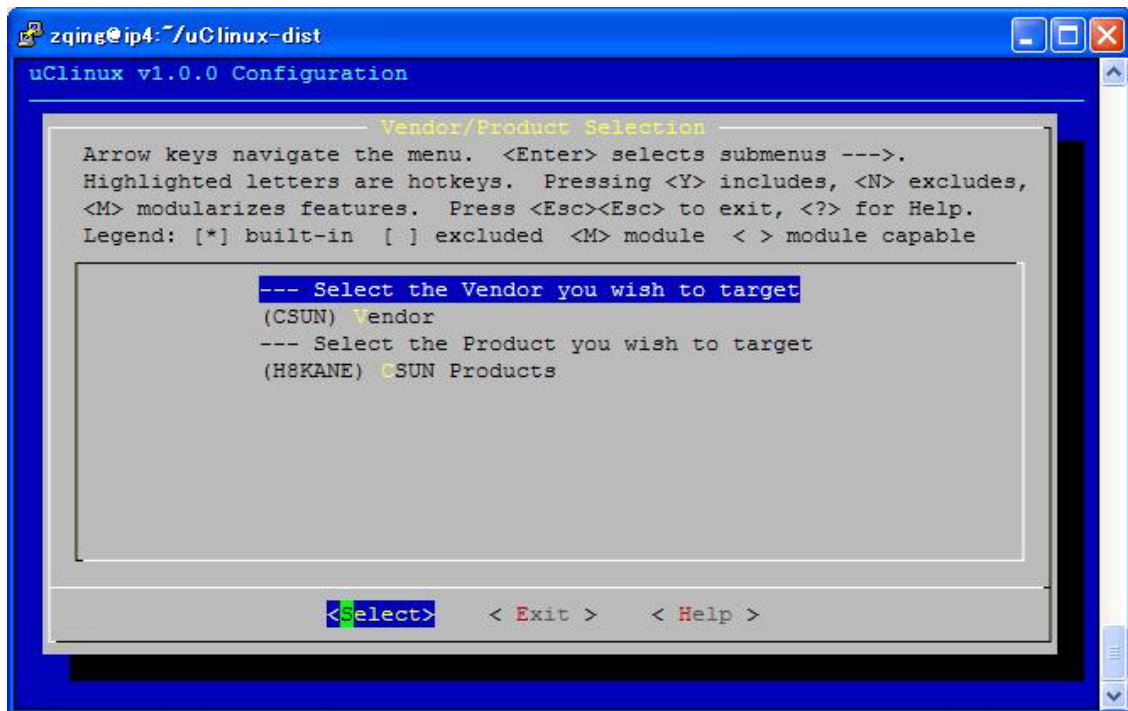
```
$make menuconfig
```

すると下の画面が開きます。

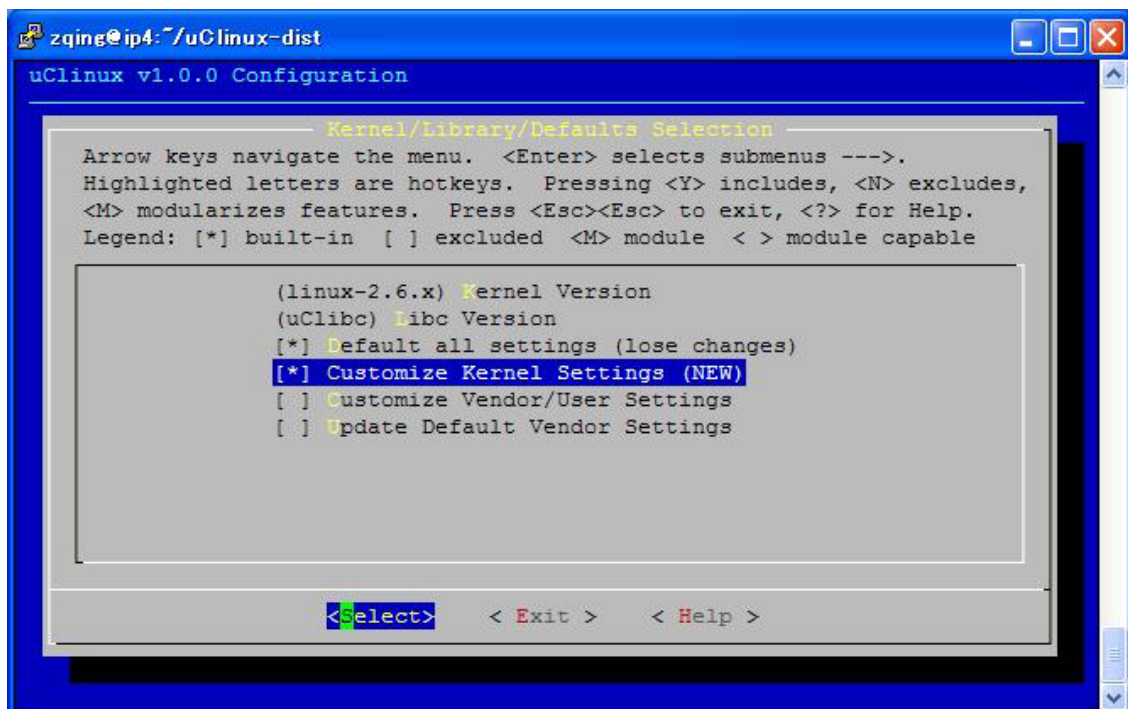


まず、「Vendor/Product Selection」を選択してください。ここで、uClinux を動作させる

プラットフォームのベンダーおよび、製品名を設定します。Vendor を「CSUN」に、Product を「H8KANE」に設定します。



画面例のように設定したら、トップメニューに戻り、「Kernel/Library/Defaults Selection」を選択してください。



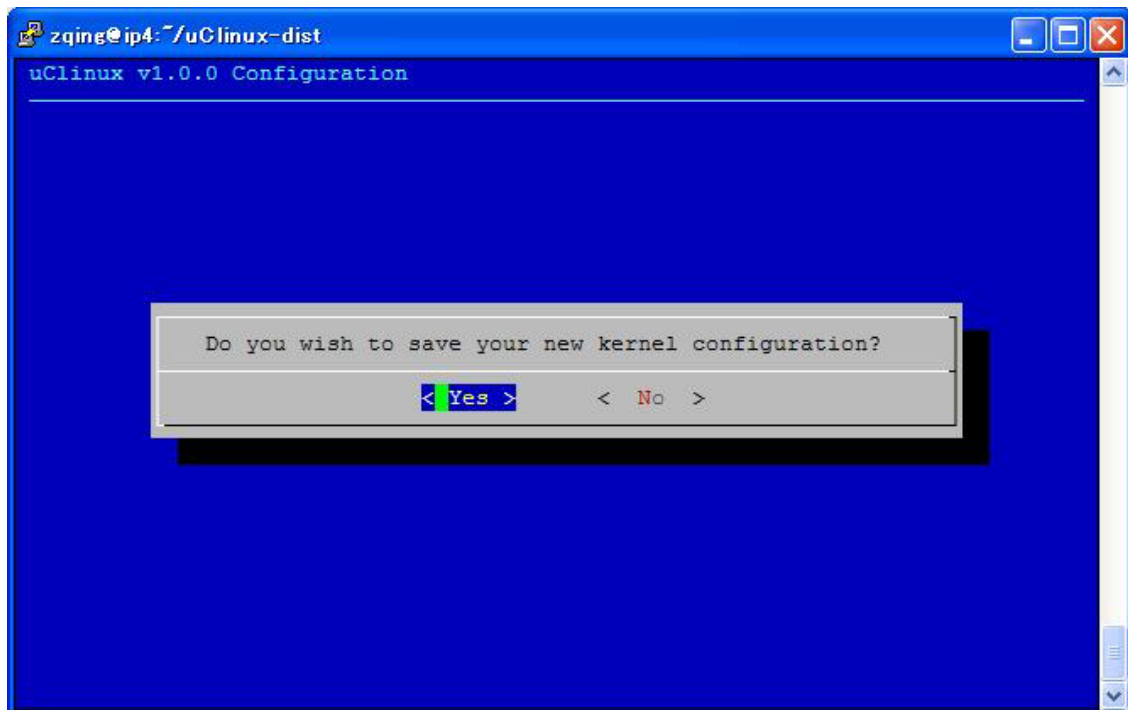
カーネルバージョンは 2.6.x を利用します。2.6.x の場合は 4MB メモリが必須です。

「Default all settings」を選択すると、uClinux-dist に同梱されているデフォルトの設定が一度、適用されます。初回のビルド時にはチェックを入れてください(2回目以降のビルド時には必要ありません)。

そして、「Customize Kernel Settings」にチェックを入れ、NFS ルートのための設定が行えるようにします。NFS は UNIX 系 OS で標準的に利用されているディスク共有です。Linux は、NFS を利用してネットワーク経由でほかのコンピュータ上のファイルシステムをルートとしてマウントし起動する、「NFS ルート」をサポートしています。NFS ルートは、H8 マイコンのように自身にルートファイルシステムとして使えるようなブロックデバイスを持たない(いわゆるディスクレス)システムで、便利に利用できます。

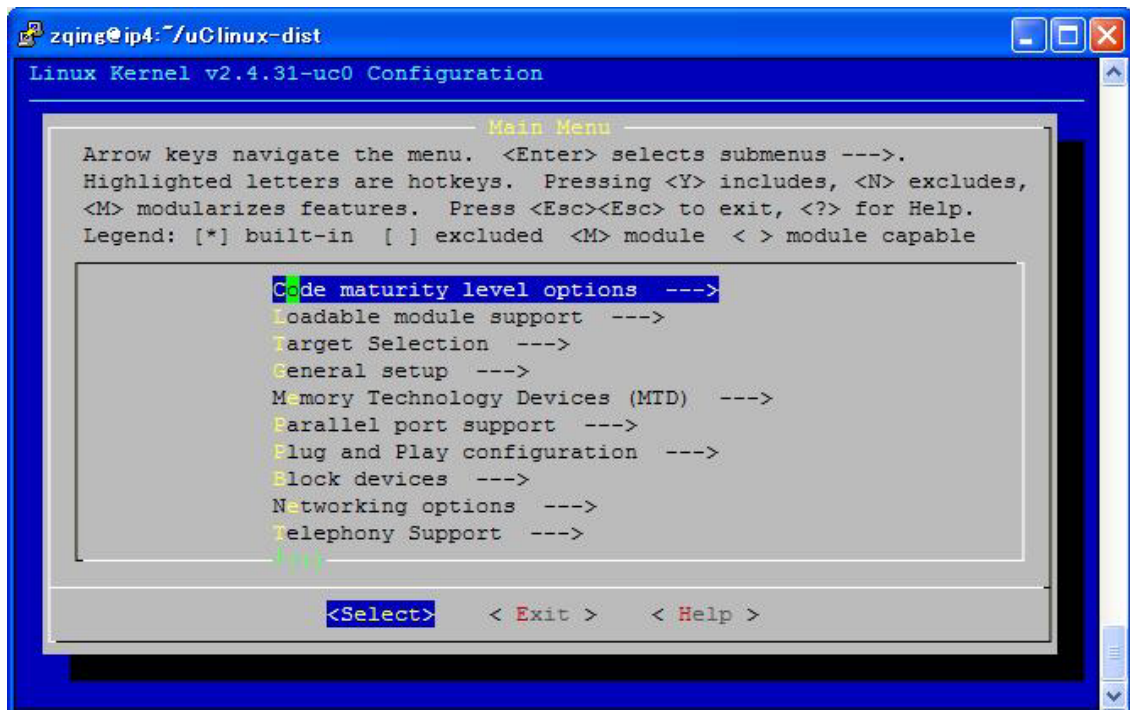
また、「Customize Kernel Settings」にチェックを入れておきましょう。

以上を設定したら、Exit を選択し、さらにトップメニューでも Exit を選択して設定の保存を指定して、メニューから抜けましょう。

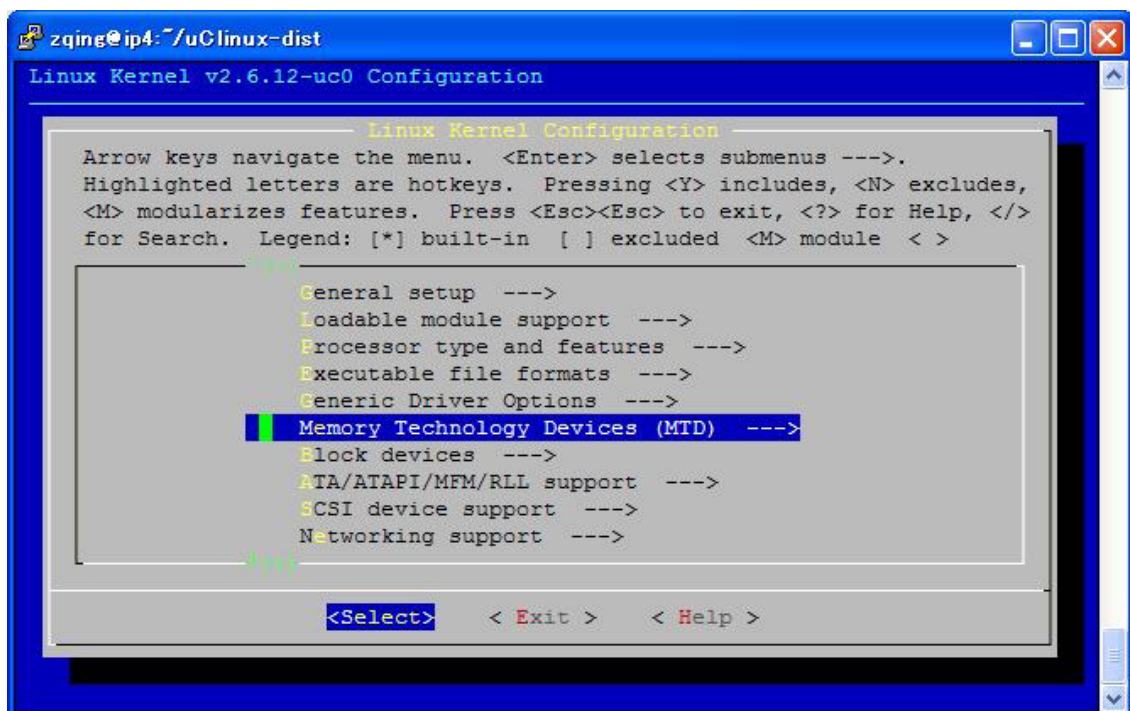


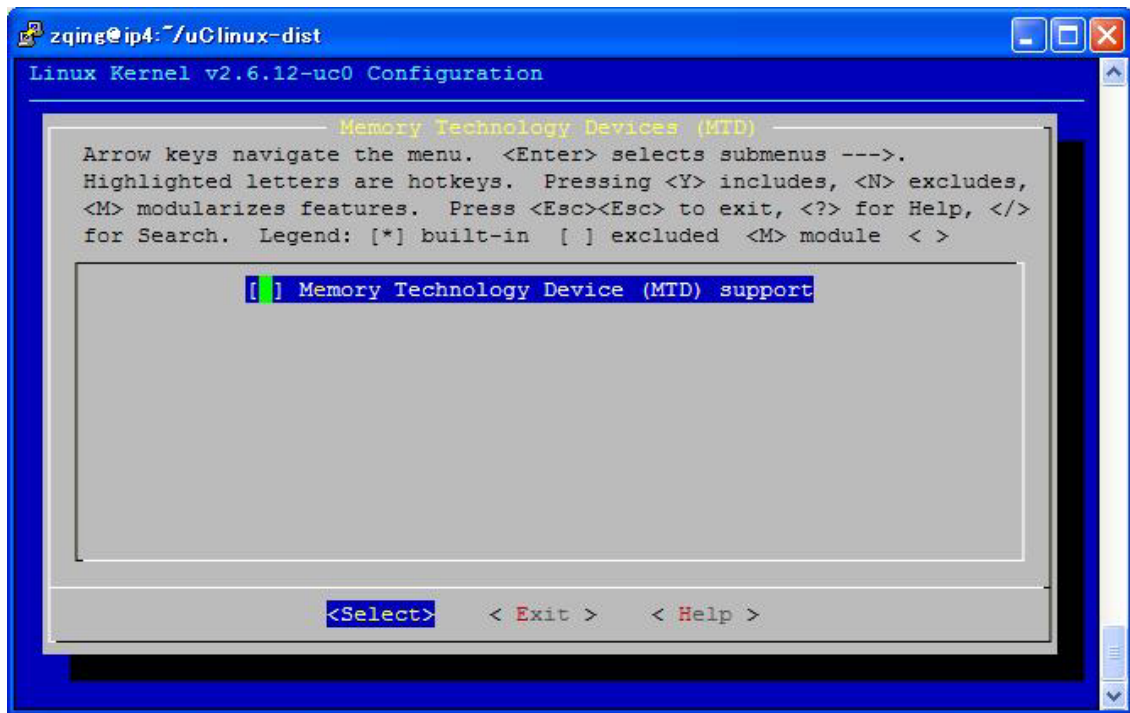
## 2. カーネル設定

ターゲットのメニューから抜けるとカーネルのデフォルト設定が適用され、自動的にカーネル設定メニューが起動してきます。

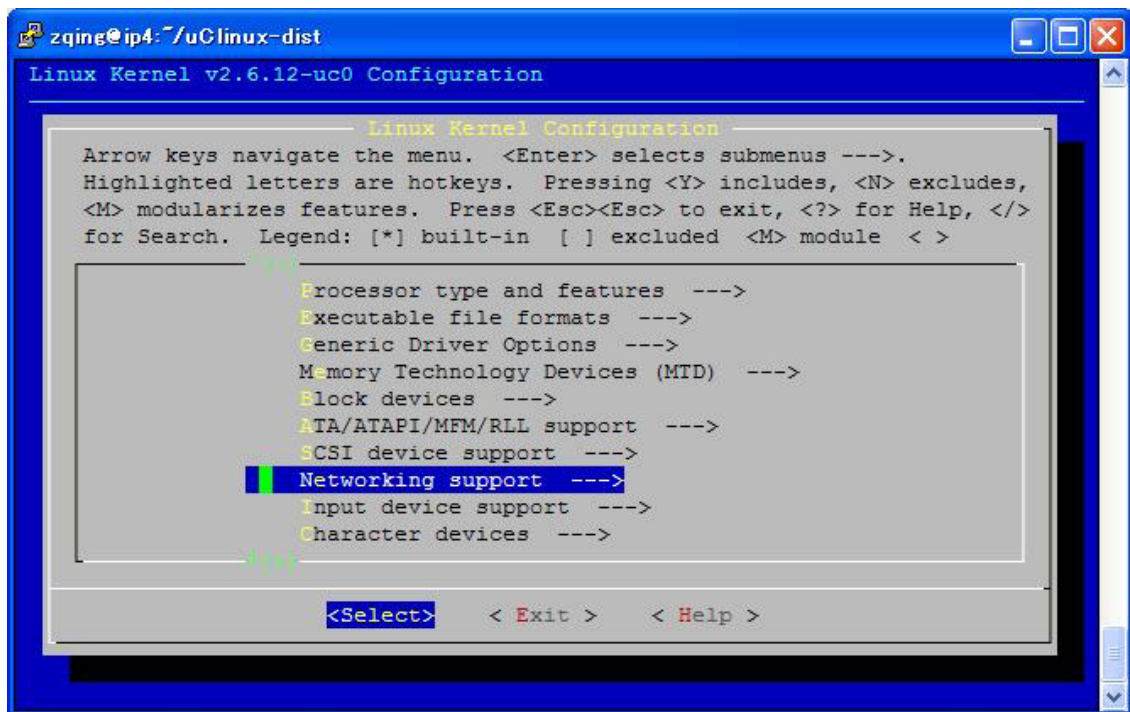


まず、「Memory Technology Devices (MTD)」を選択してください。メニューの先頭に「Memory Technology Devices (MTD) support」という項目があります。スペースキーを押してオフ(無効)にします。

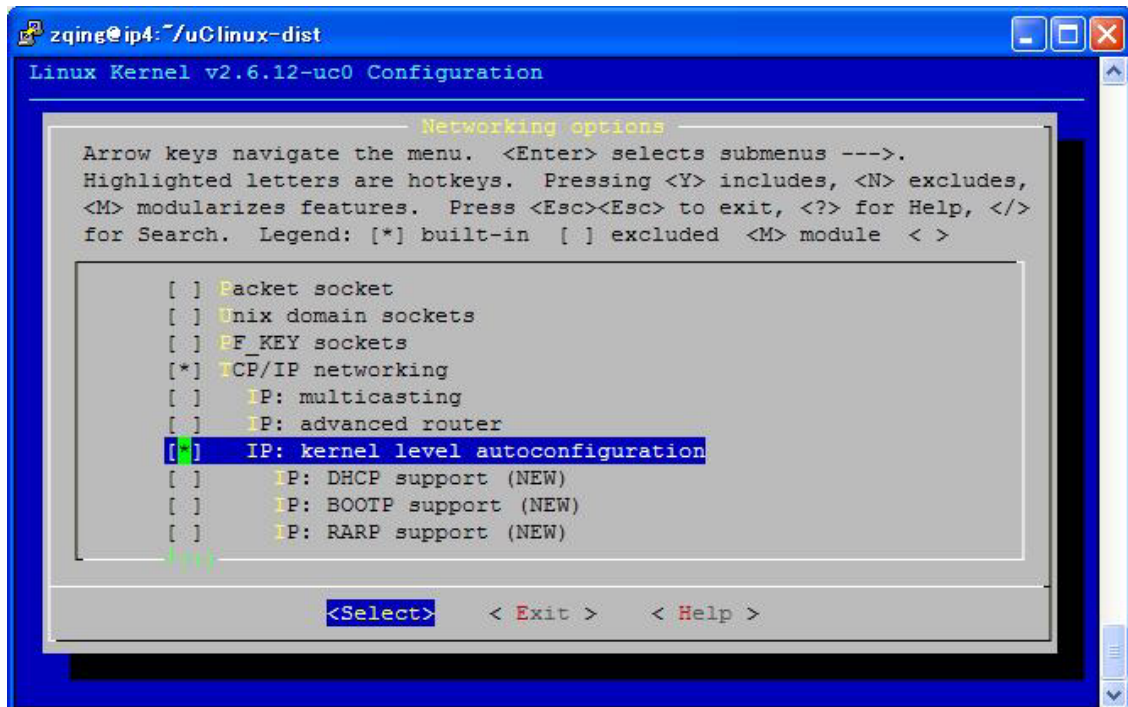
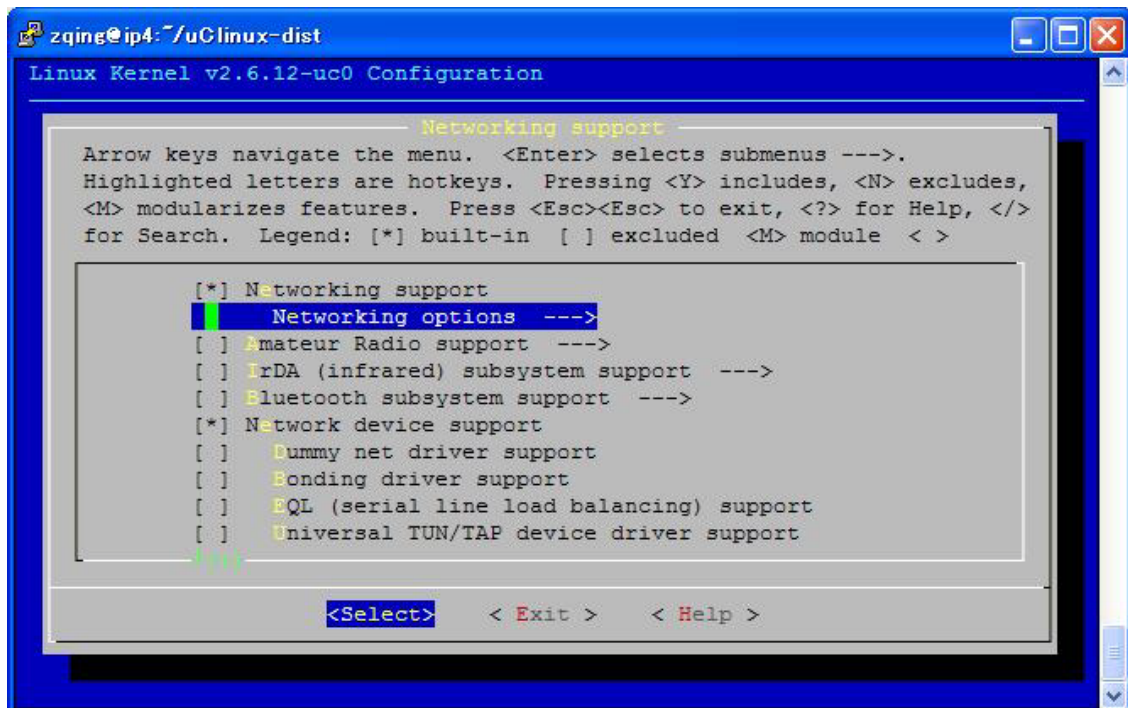




Exit を選択してカーネル設定のメニューに戻り、「Networking support」を選択します。

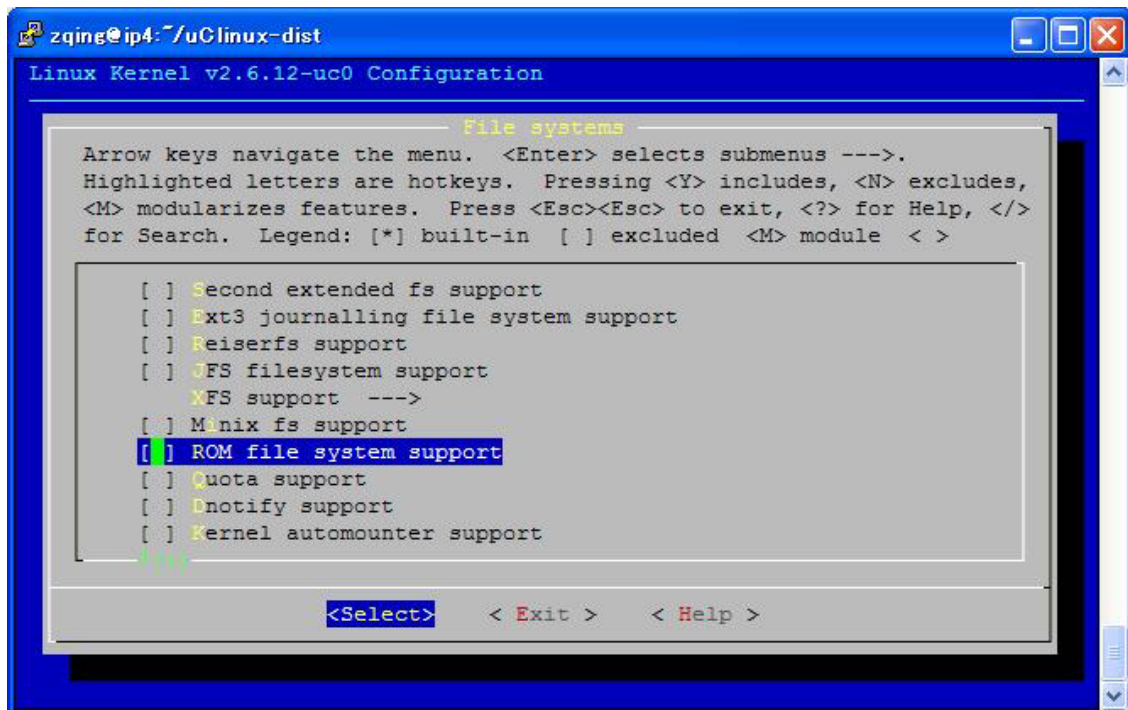


「Networking options」設定項目の中から「kernel level autoconfiguration」を探してチェックを入れます。

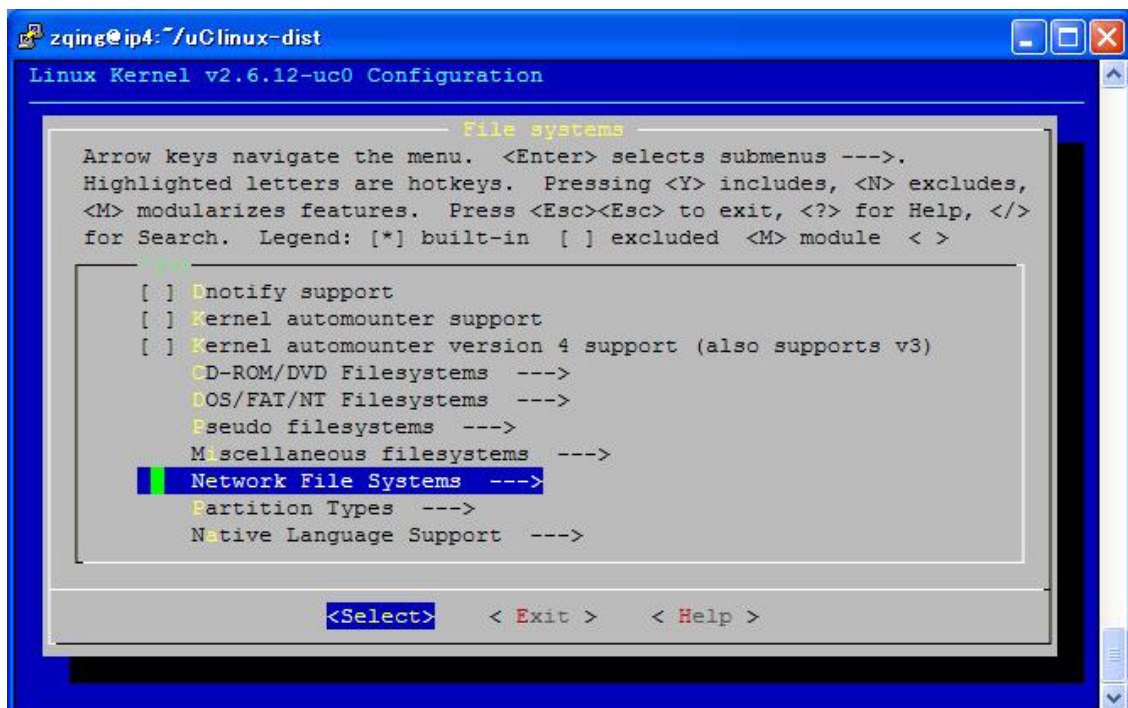


Networking options を例のように設定したら、Exit を選択してトップメニューに戻ります。File systems では、「Second extended fs support」と「Rom file system support」にチェックが入っていると思いますので、これをオフにします。

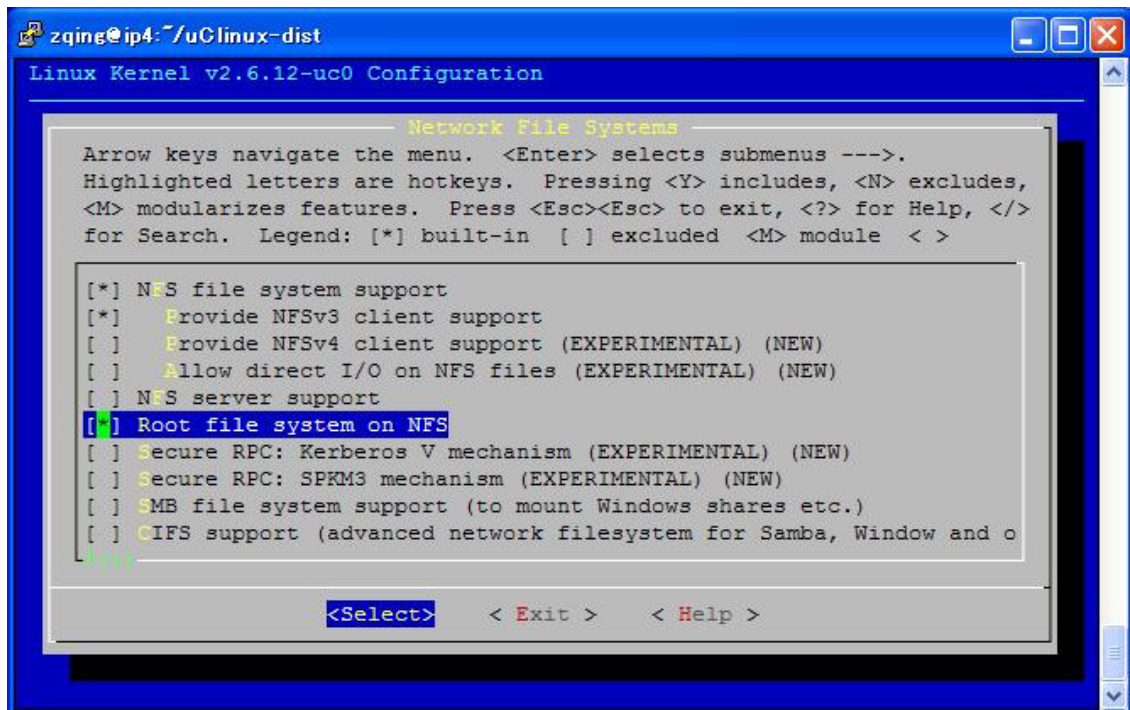




そして、File system メニューの下のほうにある「Network file Systems」という項目を選択してください。



画面例のように「NFS file system support」をオンにし、「Provide NFSv3 client support」と「Root filesystem on NFS」(NFS ルート)をオンにします。



以上で NFS ルートのカーネルの設定は完了です。Exit を選択してメニューに戻り、さらに Exit を選択してメニューを終わらせてください。終了前に設定を保存するか聞いてくるので、Yes を選択します。

### 3. ビルド

設定が保存できたら、次のコマンドを入力してビルドします。

```
$make
```

## 5.3 uClinux 環境で自作プログラムを作る

uClinux 上で動作するプログラムは、ライブラリセット uClibc が使用されています。またスタートアップファイルなども uClinux 上のものが 必要になります。従って、H8 用の gcc に多くのオプションを指定しなければなりません。オプションを手作業で入力するのは面倒なので、makefile を作成してしまったほうが良いでしょう。

簡単なプログラム「hello.c」は次のリストのとおりです。

```
#include <stdio.h>
int main(void) {
    printf("hello, uClinux-H8¥n");
}
```

```
    return 0;
}
```

Makefile は次のリストです。

```
TARGET=hello
SRCS=hello.c
```

```
CC=h8300-linux-elf-gcc
CFLAGS=-mh -mint32 -Os -fno-builtin -nostartfiles -nostdinc
LDFLAGS=-Wl, -elf2flt
STARTUP=../lib/crt0.o
INCLUDES=-I../include -I../include/include
LIBS=-L../lib -lc
```

```
OBJS=$(SRCS:.c=.o)
```

```
.c.o :
```

```
$(CC) $(CFLAGS) $(INCLUDES) $(SRCS) -c $<
```

```
$(TARGET) : $(OBJS)
```

```
$(CC) $(CFLAGS) $(INCLUDES) $(LDFLAGS) $(STARTUP) $(LIBS) $(OBJS) -o
$(TARGET)
```

この Makefile は、TARGET と SRCS を、hello/hello.c からほかの名前に変更することで、ほかの自作プログラムに流用できるよう作成しています。

uClinux-dist/myapp/を作成して、Makefile と hello.c を uClinux-dist/myapp/の下に置き、次のコマンドを実行してください。

```
$ make
```

実行ファイル hello ができるはずですので、このファイルを uClinux の NFS 又は SD ルートの下の bin(/tftpboot/nfsroot/bin)にコピーしましょう。

uClinux-H8 上では次のコマンドを入力すれば、「hello, uClinux-H8」が表示できます。



```
/> hello  
hello, uClinux-H8
```

## 5.4 アプリケーションを開発する場合の注意事項

uClinux 環境でアプリケーションを開発するのは普通の Linux とほぼ同じです。uClinux は MMU を使用しませんので、いくつが普通の Linux にはない制限があります。特に影響が大きい部分について説明します。

### fork

使えません。vfork(2)を使うようにしてください。別のプログラムを起動する場合は、単純に vfork に置き換えるだけでよいと思います。

### mmap

MAP\_ANONYMOUS だけが使用できます。ファイルアクセスの手段としてはつかえませんので、malloc+read/write を使うようにしてください。

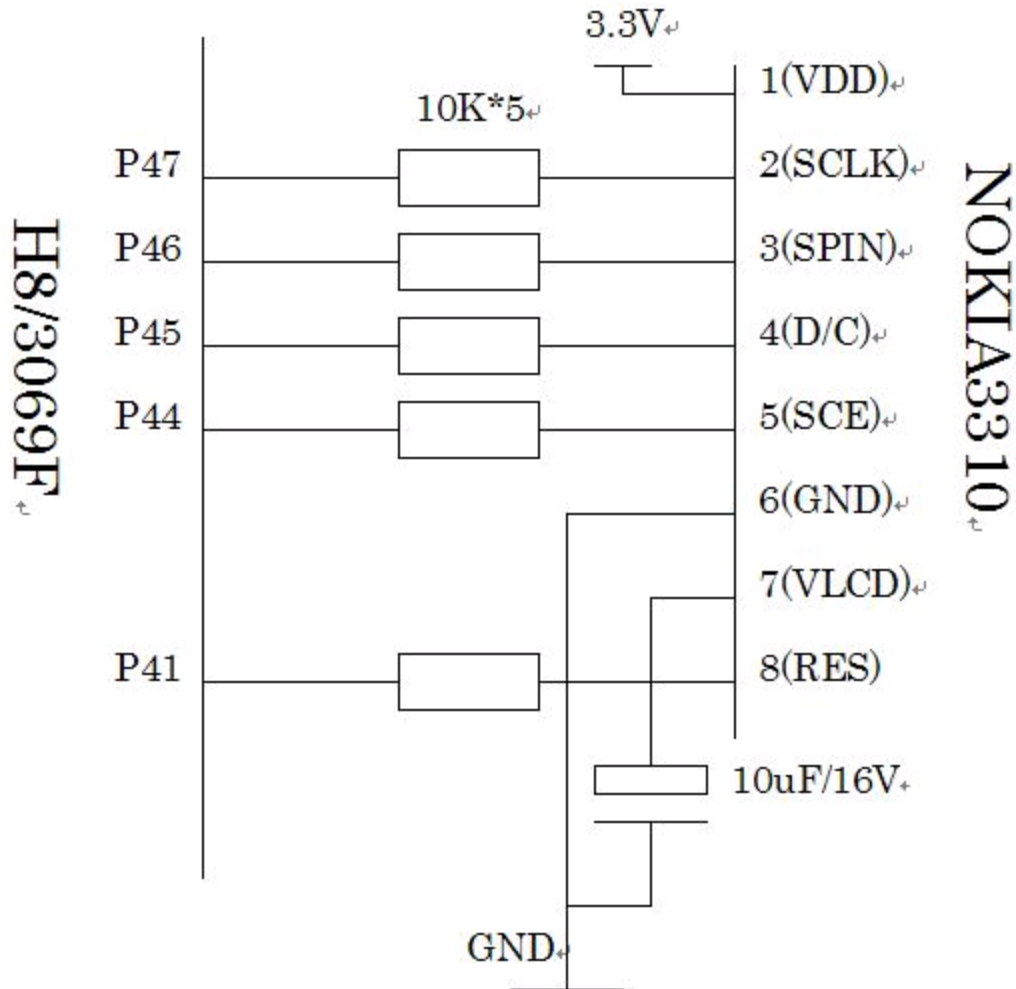
### malloc/free

頻繁に malloc/free を繰り返すと、メモリの断片化が発生してメモリ不足になることがあります。そういう処理が必要な場合は、まとめて確保してから自分で管理しましょう。

### スタック

普通の Linux では、スタック領域が不足すると自動的に拡張されますが、uClinux では指定されたサイズを固定的に割り当てるだけで、不足しでも自動的に拡張されることはありません。スタック領域の直前には BSS セクションが置かれるので、はみ出してしまうと静的変数が破壊されます。

## 5.5 液晶 NOKIA3310 を使用する



### 1. ターゲットの設定

そして次のコマンドで設定メニューを起動します。

```
$make menuconfig
```

メニューMMC/SD Card support → H8 NOKIA3310 LCD support を選択してください。

設定は完了したら、Exit を選択してメニューに戻り、さらに Exit を選択してメニューを終わらせてください。終了前に設定を保存するか聞いてくるので、Yes を選択します。

### 2. ビルド

設定が保存できたら、次のコマンドを入力してビルドします。

```
$ make
```



ビルドにはしばらく時間がかかります。コンパイル完了すれば、NOKIA3310 ドライバを含む実行ファイルを生成します。

vendors/CSUN/H8KANE/linux.bin (linux のカーネル、BIN フォーマット)

### 3. デバイスノードを作成する

uClinux のユーザーランドの/dev のしたに LCD 用のデバイスノードを作成しなければなりません。

```
mknode LCD c 201 0
```

### 4. パイプやリダイレクトをサポートするシェルに切り替える

メニュー Library/Defaults Selection → Customize Vendor/User Settings を選択してスペースキーを押し「\*」マークを入れて、Exit を選択し、メニューを終了させてください。するとユーザーランドの設定が起動します。Core Applications → Shell Program を選択して nwsh に変更します。nwsh はサイズが小さいながらもパイプやリダイレクトをサポートし、sash よりは普通のシェルに近いので操作しやすいです。

設定終わったら、make でビルドします。ビルドされた bin/nwsh を sh という名前でコピーしておきます (linux カーネルは、/bin/init → /bin/sh の順で起動します)。

nwsh シェルで、次のコマンドを入力してみます。液晶で ls の結果を表示するはずですが。

```
ls > /dev/LCD
```

### 5. LCD ドライバを使用する自作のプログラム

```
#include <stdio.h>
```

```
#define LCD_DEV "/dev/lcd"
```

```
#define BUFF_SIZE 256
```

```
#define ESC 0x1B
```

```
static char buff[BUFF_SIZE] = ``^[J(株)日昇テクノロジー¥n 多機能  
(uClinux/TOPPERS/MES)マイコンボード Kane BeBe H8/3069F(中国製)``;
```



```
int main( int argc, char *argv[] )
{
//   if( fgets( buff, BUFF_SIZE, stdin) != NULL )
    {
        FILE *fp;
        int i = 0;

        fp = fopen( LCD_DEV, "w" );
        if( fp == NULL ) return 1;

        while(1) {
            if( buff[i] == '^' ) {
                i++;
                if( buff[i] != '^' ){
                    fputc( ESC, fp );
                }
            }
            if( buff[i] == '¥0' ) break;
            if( i >= BUFF_SIZE ) break;
            fputc( buff[i++], fp );
        }
        fclose(fp);
    }
    return 0;
}
```