

# Booting an ifs image with U-boot 1.2 or 1.3 on an Freescale mpc83xx CPU#

Often boot-able ifs images are started from U-boot using the 'go' command. However, in the 1.2 and later versions of U-boot the data cache is left in an enabled and possibly locked state and the bat registers are configured in such a manner as to cause startup to fail.

As a work around an alternate bootfile, uboot.boot, can be used instead of modifying startup/lib or U-boot. This boot file will disable the data cache and clear the bat registers before jumping to startup.

For more information about bootfiles see the `mkifs` documentation in the *Utilities Reference*.

## Compiling the uboot.boot bootfile #

The uboot.boot bootfile can be recompiled from `uboot.s` by the following steps:

```
qcc -Vgcc_ntoppc -c -O -Wa,-I. -Wa,--defsym -Wa,VARIANT_be=1 -EB uboot.s
qcc -Vgcc_ntoppc -Wl,--no-keep-memory -Wl,-Ttext -Wl,0x0 -nostartup -nostdlib -ouboot.boot uboot.o -EB
rm uboot.o
```

## Build a bootable ifs image using the uboot.boot bootfile #

### Command Line#

- Ensure that uboot.boot is copied to `install/ppcbe/boot/sys/` under the BSP root directory, or an alternate suitable location, such as `prebuilt/ppcbe/boot/sys`
- Edit the build file to set the `bootfile_name` of the virtual attribute of the `.bootstrap` file to uboot:

```
[virtual=ppcbe,uboot] .bootstrap = {
...
}
```

- Build the ifs image either by using `make` in the images directory, or by running `mkifs -r ../install <build file> <ifs image>` e.g.

```
mkifs -r ../install mpc8349e-mds.build ifs-8349mds.uboot
```

### Momentics IDE#

- In the *QNX System Builder* window select the image.
- Change the `system->boot file` property to `uboot` by selecting the `value` field and typing `uboot`. (`uboot` will not be on the drop down list.)
- Rebuild the ifs image, perhaps by selecting the `project->build project` menu item.

## Using U-boot to Boot a bootable ifs image with the uboot.boot bootfile #

First load the ifs image in U-boot using the standard method, such as:

```
=> tftpboot 0x200000 <ifs image>
```

e.g.

```
=> tftpboot 0x200000 ifs-8349mds.uboot
```

Then to jump to the start of the ifs image, in U-boot do:

```
=> go 0x200000
```

An example:

U-Boot 1.3.0-rc3 (Nov 6 2007 - 09:38:18) MPC83XX

Reset Status:

CPU: e300c1, MPC8349E, Rev: 11 at 528 MHz, CSB: 264 MHz

Board: Freescale MPC8349EMDS

I2C: ready

DRAM: 256 MB (DDR1, 64-bit, ECC off)

FLASH: 8 MB

In: serial

Out: serial

Err: serial

Net: TSEC0, TSEC1

Type run flash\_nfs to mount root filesystem over NFS

Hit any key to stop autoboot: 0

```
=> tftpboot 0x200000 ifs-8349mds.uboot
```

Speed: 100, full duplex

Using TSEC0 device

TFTP from server 192.168.0.10; our IP address is 192.168.0.5

Filename 'ifs-8349mds.uboot'.

Load address: 0x200000

Loading: #####

done

Bytes transferred = 646364 (9dcdc hex)

```
=> go 0x200000
```

```
## Starting application at 0x00200000 ...
```

System page at phys:0000c000 user:0000c000 kern:0000c000

Starting next program at v0023b240

Welcome to QNX Neutrino 6.3 on the Freescale MPC8349 MDS board

#

## Bootfile Example uboot.s#

.text

```
.set STARTUP_ENTRY, 12
```

```
.set HID0, 0x3f0
```

```
.equ IBAT0L, 0x211
```

```
.equ IBAT0U, 0x210
```

```
.equ IBAT1L, 0x213
```

```
.equ IBAT1U, 0x212
```

```
.equ IBAT2L, 0x215
```

```
.equ IBAT2U, 0x214
```

```
.equ IBAT3L, 0x217
```

```
.equ IBAT3U, 0x216
```

```
.equ IBAT4L, 0x231
```

```
.equ IBAT4U, 0x230
```

```
.equ IBAT5L, 0x233
.equ IBAT5U, 0x232
.equ IBAT6L, 0x235
.equ IBAT6U, 0x234
.equ IBAT7L, 0x237
.equ IBAT7U, 0x236
```

```
.equ DBAT0L, 0x219
.equ DBAT0U, 0x218
.equ DBAT1L, 0x21B
.equ DBAT1U, 0x21A
.equ DBAT2L, 0x21D
.equ DBAT2U, 0x21C
.equ DBAT3L, 0x21F
.equ DBAT3U, 0x21E
.equ DBAT4L, 0x239
.equ DBAT4U, 0x238
.equ DBAT5L, 0x23B
.equ DBAT5U, 0x23A
.equ DBAT6L, 0x23D
.equ DBAT6U, 0x23C
.equ DBAT7L, 0x23F
.equ DBAT7U, 0x23E
```

```
.globl _start
```

```
_start:
```

```
.ascii "["
.ascii " default_image=0x1f0000"
.ascii " attr=\"?+bigendian\""
.ascii " len=0x100"
.asciz "]"
```

```
# Align to 4 bytes
.byte 0
```

```
.align 2
.ascii "boot"
```

```
bl 1f # get load base address (+4)
```

```
1:
```

```
# clear all bats, but 0
# - we assume that the DDR is handled by [DI]BAT0[UL]
```

```
xor %r0, %r0, %r0
mtspr DBAT1U, %r0
mtspr DBAT1L, %r0
mtspr DBAT2U, %r0
mtspr DBAT2L, %r0
mtspr DBAT3U, %r0
mtspr DBAT3L, %r0
mtspr DBAT4U, %r0
mtspr DBAT4L, %r0
mtspr DBAT5U, %r0
mtspr DBAT5L, %r0
mtspr DBAT6U, %r0
mtspr DBAT6L, %r0
```

```
mtspr DBAT7U, %r0
mtspr DBAT7L, %r0
mtspr IBAT1U, %r0
mtspr IBAT1L, %r0
mtspr IBAT2U, %r0
mtspr IBAT2L, %r0
mtspr IBAT3U, %r0
mtspr IBAT3L, %r0
mtspr IBAT4U, %r0
mtspr IBAT4L, %r0
mtspr IBAT5U, %r0
mtspr IBAT5L, %r0
mtspr IBAT6U, %r0
mtspr IBAT6L, %r0
mtspr IBAT7U, %r0
mtspr IBAT7L, %r0
sync
```

# disable data cache, turn off lock

```
mfspr %r3, HID0
lis %r4, 0
ori %r4, %r4, (1 << 14) | (1 << 12) #HID0_DCE|HID0_DLOCK
andc %r3, %r3, %r4
ori %r4, %r3, (1 << 10) # HID0_DCI
sync
mtspr HID0, %r4 # sets invalidate, clears enable and lock
sync
mtspr HID0, %r3 # clears invalidate
```

# jump into startup

```
mflr %r31
lwz %r31, STARTUP_ENTRY+0x100-4(%r31) # get start addr
mtlr %r31
blr # transfer into startup
```