

i.MX7 SABRE SD Board—Running the Core at 1.2 GHz

1. Introduction

NXP introduced a new part to the i.MX7 D lineup of MPUs. These new MPUs (MCIMX7D2DVK12SC and MCIMX7D2DVM12SC) can run their cores up to a maximum of 1.2 GHz. This application note shows you how to run the current SABRE SD board at this higher frequency.

Note that the components on the SABRE board are specified to a maximum core frequency of 1.0 GHz. Running at this higher frequency is not guaranteed to function and reduces the lifetime of the part. See *i.MX7 Dual/Solo Product Lifetime Usage* (document [AN5334](#)) for details.

Although the i.MX7 that is delivered on this board is not specified to run at this higher frequency, most parts tolerate this frequency at a room temperature for evaluation purposes.

The two new MPUs added to the i.MX 7D family (MCIMX7D2DVK12SC and MCIMX7D2DVM12SC) are the only parts guaranteed to operate at 1.2 GHz. They have no reduction of operating lifetime at this CPU frequency.

Contents

1. Introduction	1
2. Required operational changes	2
2.1. BSP changes	2
3. Required u-boot changes	2
4. Verifying faster speeds	3
5. Getting the patch	5
6. Revision History	6



2. Required operational changes

For the core to operate at this higher frequency, the Vdd_ARM voltage must be increased to 1.22 V from 1.1 V (1 GHz). Also, the clock driving the ARM® Cortex®-A7 cores must be increased to 1.2 GHz. These parameters are software-controlled on the SABRE board. It means that the higher speed evaluation is only available while running Linux with a modified DTB file.

2.1. BSP changes

The changes to the BSP are very simple and limited to one file. A new DTB file is required to enable Linux to change the frequency and the Vdd_ARM voltage. The source file *imx7d.dtsi* must be modified and then recompiled.

An additional line must be inserted into the “operating-points” section (highlighted in yellow). The diff file output is as follows:

```
diff --git a/arch/arm/boot/dts/imx7d.dtsi b/arch/arm/boot/dts/imx7d.dtsi
index 5053a7d..014ea01 100644
--- a/arch/arm/boot/dts/imx7d.dtsi
+++ b/arch/arm/boot/dts/imx7d.dtsi
@@ -53,6 +53,7 @@
                reg = <0>;
                operating-points = <
                    /* KHz uV */
+               1200000 1220000
                    996000 1075000
                    792000 975000
                >;
```

The added line adds 1200000 KHz and 1200000 μ V, which indicates a new operational point of 1.2 GHz at 1.2 V. The higher voltage enables the chip to operate at this higher frequency. Note that this higher voltage reduces the operating life of the chip, so run at this frequency/voltage combination only when needed. Linux changes the operating frequency up and down as it determines the need for additional performance.

When this change is made to the DTSI file and the file is recompiled, it generates the proper DTB file that must be included in the u-boot environment, and this information is passed to the Linux kernel when booting.

3. Required u-boot changes

For Linux to operate at this new speed, add the new file to the bootable partition and make the “fdt_file” parameter point to the new DTB file. To accomplish this, copy the new DTB file to the FAT file system side of the SD card. Change the “fdt_file” parameter to point to the new file, as shown in [Figure 1](#).

In this example, the DTB file is named “imx7d-sdb.1.2ghz.dtb”:

```
=> setenv fdt_file imx7d-sdb.1.2ghz.dtb
=> 
```

Figure 1. DTB file naming

This is confirmed as follows:

```
Environment size: 2256/8188 bytes
=> setenv fdt_file imx7d-sdb.1.2ghz.dtb
=> printenv
baudrate=115200
boot_fdt=try
bootcmd=mmc dev ${mmcdev}; mmc dev ${mmcdev}; if mmc rescan; then if run loadbootscript; then run bootscript; else if run loadimage; then
run mmcboot; else run netboot; fi; fi; else run netboot; fi
bootcmd_mfg=run mfgtool_args; bootz ${loadaddr} ${initrd_addr} ${fdt_addr};
bootdelay=3
bootscript=echo Running bootscript from mmc ...; source
console=ttymxc0
dfu_alt_info=image raw 0 0x800000;u-boot raw 0 0x4000;bootimg part 0 1;rootfs part 0 2
ethact=FEC0
ethaddr=00:04:9f:04:4b:3c
ethprime=FEC
fdt_addr=0x83000000
fdt_file=imx7d-sdb.1.2ghz.dtb
fdt_high=0xffffffff
image=zImage
initrd_addr=0x83800000
initrd_high=0xffffffff
ip_dyn=yes
loadaddr=0x80800000
loadbootscript=fatload mmc ${mmcdev}:${mmcpart} ${loadaddr} ${script};
loadfdt=fatload mmc ${mmcdev}:${mmcpart} ${fdt_addr} ${fdt_file}
loadimage=fatload mmc ${mmcdev}:${mmcpart} ${loadaddr} ${image}
loadmimage=fatload mmc ${mmcdev}:${mmcpart} 0x7F8000 ${mimage}
m1boot=run loadmimage; bootaux 0x7F8000
m1image=m1_qspi.bin
mfgtool_args=setenv bootargs console=${console},${baudrate} rdinit=/linuxrc g_mass_storage.stall=0 g_mass_storage.removable=1 g_mass_stora
ge.idVendor=0x066F g_mass_storage.idProduct=0x37FF g_mass_storage.isSerialNumber="" clk_ignore_unused
mmcargs=setenv bootargs console=${console},${baudrate} root=${mmcrroot}
mmcautoloadetect=yes
mmcboot=echo Booting from mmc ...; run mmcargs; if test ${boot_fdt} = yes || test ${boot_fdt} = try; then if run loadfdt; then bootz ${loa
daddr} - ${fdt_addr}; else if test ${boot_fdt} = try; then bootz; else echo WARN: Cannot load the DT; fi; fi; else bootz; fi;
mmcdev=0
mmcpart=1
mmcrroot=/dev/mmcblk0p2 rootwait rw
netargs=setenv bootargs console=${console},${baudrate} root=/dev/nfs ip=dhcp nfsroot=${serverip}:${nfsroot},v3,tcp
netboot=echo Booting from net ...; run netargs; if test ${ip_dyn} = yes; then setenv get_cmd dhcp; else setenv get_cmd tftp; fi; ${get_cmd
} ${image}; if test ${boot_fdt} = yes || test ${boot_fdt} = try; then if ${get_cmd} ${fdt_addr} ${fdt_file}; then bootz ${loadaddr} - ${fd
t_addr}; else if test ${boot_fdt} = try; then bootz; else echo WARN: Cannot load the DT; fi; fi; else bootz; fi;
panel=TF143AB
script=boot.scr
Environment size: 2263/8188 bytes
=>
```

Figure 2. DTB file naming confirmation

4. Verifying faster speeds

To verify that the new FDT file is used, run the *cpufreq-info* program on the serial console. Here is the output you should see:

```
root@imx7dsabresd:~# cpufreq-info
cpufrequtils 008: cpufreq-info (C) Dominik Brodowski 2004-2009
Report errors and bugs to cpufreq@vger.kernel.org, please.
analyzing CPU 0:
driver: imx7d-cpufreq
CPUs which run at the same hardware frequency: 0 1
CPUs which need to have their frequency coordinated by software: 0 1
maximum transition latency: 61.0 us.
hardware limits: 792 MHz - 1.20 GHz
available frequency steps: 792 MHz, 996 MHz, 1.20 GHz
available cpufreq governors: interactive, conservative, userspace, powersave, ondemand, performance
current policy: frequency should be within 792 MHz and 1.20 GHz
The governor "ondemand" may decide which speed to use
within this range.
current CPU frequency is 792 MHz (asserted by call to hardware).
cpufreq stats: 792 MHz:99.41%, 996 MHz:0.30%, 1.20 GHz:0.28% (56)
analyzing CPU 1:
driver: imx7d-cpufreq
CPUs which run at the same hardware frequency: 0 1
CPUs which need to have their frequency coordinated by software: 0 1
maximum transition latency: 61.0 us.
hardware limits: 792 MHz - 1.20 GHz
available frequency steps: 792 MHz, 996 MHz, 1.20 GHz
available cpufreq governors: interactive, conservative, userspace, powersave, ondemand, performance
current policy: frequency should be within 792 MHz and 1.20 GHz.
The governor "ondemand" may decide which speed to use
within this range.
current CPU frequency is 792 MHz (asserted by call to hardware).
cpufreq stats: 792 MHz:99.41%, 996 MHz:0.30%, 1.20 GHz:0.28% (56)
root@imx7dsabresd:~#
```

Figure 3. Verifying faster speeds

This program reports that the additional frequency of 1.2 GHz is available for the Linux kernel to be selected, based on the CPU load.

Now that it is verified that the new speed is available, run the Dhrystone benchmark to see the difference in speed. The screenshots indicate that there is a performance difference of 20 % (which is expected). Note that the key item here is the difference and not the raw number.

This is the 1 GHz (original) DTB file:

```
root@imx7dsabresd:~# ./dry2

Dhrystone Benchmark, Version C, Version 2.2
Program compiled without 'register' attribute
Using times(), HZ=100

Trying 50000 runs through Dhrystone:
Measured time too small to obtain meaningful results

Trying 500000 runs through Dhrystone:
Measured time too small to obtain meaningful results

Trying 5000000 runs through Dhrystone:
Measured time too small to obtain meaningful results

Trying 50000000 runs through Dhrystone:
Microseconds for one run through Dhrystone: 0.3
Dhrystones per Second: 3770739

root@imx7dsabresd:~# █
```

Figure 4. 1 GHz DTB file

This is the 1.2 GHz DTB file:

```
root@imx7dsabresd:~# ./dry2

Dhrystone Benchmark, Version C, Version 2.2
Program compiled without 'register' attribute
Using times(), HZ=100

Trying 50000 runs through Dhrystone:
Measured time too small to obtain meaningful results

Trying 500000 runs through Dhrystone:
Measured time too small to obtain meaningful results

Trying 5000000 runs through Dhrystone:
Measured time too small to obtain meaningful results

Trying 50000000 runs through Dhrystone:
Microseconds for one run through Dhrystone: 0.2
Dhrystones per Second: 4541326

root@imx7dsabresd:~# █
```

Figure 5. 1.2 GHz DTB file

When running the Dhrystone benchmark, run `cpufreq-info`. Figure 6 shows that the core is running at 1.2 GHz:

```

root@imx7dsabresd:~# ./dry2 &
[1] 888

Dhrystone Benchmark, Version C, Version 2.2
Program compiled without 'register' attribute
Using times(), HZ=100

Trying 50000 runs through Dhrystone:
root@imx7dsabresd:~# Measured time too small to obtain meaningful results

Trying 500000 runs through Dhrystone:
Measured time too small to obtain meaningful results

Trying 5000000 runs through Dhrystone:
Measured time too small to obtain meaningful results

Trying 50000000 runs through Dhrystone:
cpufreq-info
cpufrequtils 008: cpufreq-info (C) Dominik Brodowski 2004-2009
Report errors and bugs to cpufreq@vger.kernel.org, please.
analyzing CPU 0:
  driver: imx7d-cpufreq
  CPUs which run at the same hardware frequency: 0 1
  CPUs which need to have their frequency coordinated by software: 0 1
  maximum transition latency: 61.0 us.
  hardware limits: 792 MHz - 1.20 GHz
  available frequency steps: 792 MHz, 996 MHz, 1.20 GHz
  available cpufreq governors: interactive, conservative, userspace, powersave, ondemand, performance
  current policy: frequency should be within 792 MHz and 1.20 GHz.
    The governor "ondemand" may decide which speed to use
    within this range.
  current CPU frequency is 1.20 GHz (asserted by call to hardware).
  cpufreq stats: 792 MHz:99.44%, 996 MHz:0.06%, 1.20 GHz:0.50% (92)
analyzing CPU 1:
  driver: imx7d-cpufreq
  CPUs which run at the same hardware frequency: 0 1
  CPUs which need to have their frequency coordinated by software: 0 1
  maximum transition latency: 61.0 us.
  hardware limits: 792 MHz - 1.20 GHz
  available frequency steps: 792 MHz, 996 MHz, 1.20 GHz
  available cpufreq governors: interactive, conservative, userspace, powersave, ondemand, performance
  current policy: frequency should be within 792 MHz and 1.20 GHz.
    The governor "ondemand" may decide which speed to use
    within this range.
  current CPU frequency is 1.20 GHz (asserted by call to hardware).
  cpufreq stats: 792 MHz:99.44%, 996 MHz:0.06%, 1.20 GHz:0.50% (92)
root@imx7dsabresd:~# Microseconds for one run through Dhrystone:          0.2
Dhrystones per Second:          4545454

```

Figure 6. cpufreq-info

5. Getting the patch

The patch is attached to this application note.

NOTE

This DTB file works with the NXP Linux release version 4.1.15_1.2.0.

6. Revision History

Table 1 summarizes the changes done to this document since the initial release.

Table 1. Revision history

Revision number	Date	Substantive changes
0	04/2017	Initial release

How to Reach Us:

Home Page:

nxp.com

Web Support:

nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address:

nxp.com/SalesTermsandConditions.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, Freescale, and the Freescale logo are trademarks of NXP B.V. All other product or service names are the property of their respective owners.

ARM, the ARM Powered logo, and Cortex are registered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved.

© 2017 NXP B.V.

Document Number: AN11957

Rev. 0

04/2017

