Cross Compiling WiFi Driver & Tools For Android

1. Enviroment

- **Host**
  ubuntu 20.04 LTS
- **Target**
  i.MX8QM-MEK & Android 11.0.0_2.6.0
- **Driver Version**

PCIE-WLAN-UART-BT-8997-U16-X86-16.92.21.p55.3-16.92.21.p55.3-MM5X16344.P3-MGPL

- **WiFi Chip**
  88W8997

2. Cross Compiling Android image for i.MX8QM-MEK

(1) Preparing cross compiler

- **gcc-arm-8.3-2019.03-x86_64-aarch64-elf.tar.xz**
  https://developer.arm.com/downloads/-/gnu-a
  Then:

```
# sudo tar -xvJf gcc-arm-8.3-2019.03-x86_64-aarch64-linux-gnu.tar.xz-C /opt
# export AARCH64_GCC_CROSS_COMPILE=/opt/gcc-arm-8.3-2019.03-x86_64-aarch64-linuxgnu/bin/aarch64-linux-gnu-
```

- **android-clang**

```
# sudo git clone https://android.googlesource.com/platform/prebuilts/clang/host/linux-x86 /opt/prebuilt-android-clang -b master-kernel-build-2021
# cd /opt/prebuilt-android-clang
# sudo git checkout bceb7274dda5bb587a5473058bd9f52e678dde98
# export CLANG_PATH=/opt/prebuilt-android-clang
```

(2) Cross Compiling Android image

Before cross compiling WiFi driver, android image for target board must be built. Because we need kernel directory for cross compiling WiFi driver.

All steps are described in Android_User's_Guide.pdf, So these steps are omitted here.

3. Cross Compiling WiFi Driver & Tools

(1) Driver & Tools directories

The WiFi driver source code in directory '**wlan_src**', and in it, mapp directory is source code of tools. These tools will be built together with WiFi driver.

- **In wlan_src**

```
Makefile    mapp    mlan.mod    moal.mod    README_MLAN    README_UAP
script
gpl-2.0.txt    mlan    mlinux    README    README_RBC    README_WIFIDIRECT
```

- **In wlan_src/mapp**

```
mlan2040coex    mlanconfig    mlanevent    mlanutl    uaputl    wifidirectutl
```

(2) Updating Makefiles

- wlan_src/Makefile

```
--- Makefile   2022-12-04 18:46:17.820929821 -0800
+++ Makefile-update         2022-12-04 18:51:47.390364081 -0800
@@ -19,12 +19,13 @@
 CONFIG_COMPATDIR=n
 ifeq ($(CONFIG_COMPATDIR), y)
 COMPATDIR=/lib/modules/$(KERNELVERSION_X86)/build/compat-wireless-3.2-rc1-1/include
-CC =            $(CROSS_COMPILE)gcc -I$(COMPATDIR)
+CC ?=           $(CROSS_COMPILE)gcc -I$(COMPATDIR)
 else
-CC =            $(CROSS_COMPILE)gcc
+CC ?=           $(CLANG_CC)
+CXX ?=          $(CLANG_CXX)
 endif

-LD ?=           $(CROSS_COMPILE)ld
+LD ?=           $(LLVM_LD)
 BACKUP=         /root/backup
 YMD=            `date +%Y%m%d%H%M`

@@ -104,11 +105,11 @@



-CONFIG_ANDROID_KERNEL=n
+CONFIG_ANDROID_KERNEL=y


 #32bit app over 64bit kernel support
-CONFIG_USERSPACE_32BIT_OVER_KERNEL_64BIT=n
+CONFIG_USERSPACE_32BIT_OVER_KERNEL_64BIT=y


 ##############################################################################
@@ -325,14 +326,14 @@
 endif

 # add -Wno-packed-bitfield-compat when GCC version greater than 4.4
-GCC_VERSION := $(shell echo `gcc -dumpversion | cut -f1-2 -d.` \>= 4.4 | sed -e 's/\./*100+/g' | bc )
-ifeq ($(GCC_VERSION),1)
-       ccflags-y += -Wno-packed-bitfield-compat
-endif
-WimpGCC_VERSION := $(shell echo `gcc -dumpversion | cut -f1 -d.`| bc )
-ifeq ($(shell test $(WimpGCC_VERSION) -ge 7; echo $$?),0)
-ccflags-y += -Wimplicit-fallthrough=3
-endif
+#GCC_VERSION := $(shell echo `gcc -dumpversion | cut -f1-2 -d.` \>= 4.4 | sed -e 's/\./*100+/g' | bc )
+#ifeq ($(GCC_VERSION),1)
+#      ccflags-y += -Wno-packed-bitfield-compat
+#endif
+#WimpGCC_VERSION := $(shell echo `gcc -dumpversion | cut -f1 -d.`| bc )
+#ifeq ($(shell test $(WimpGCC_VERSION) -ge 7; echo $$?),0)
+#ccflags-y += -Wimplicit-fallthrough=3
+#endif
 #ccflags-y += -Wunused-but-set-variable
 #ccflags-y += -Wmissing-prototypes
 #ccflags-y += -Wold-style-definition
```

- wlan_src/mapp/wifidirectutl/Makefile

```
--- Makefile        2022-08-29 09:50:14.000000000 -0700
+++ Makefile-update    2022-12-04 18:55:06.403230202 -0800
@@ -21,7 +21,11 @@

 CFLAGS += -Wall
 ifeq (,$(findstring ANDROID_KERNEL, $(CFLAGS)))
-LIBS=-lrt
+       LIBS = -lrt
+else
+       LIBS += -L/opt/prebuilt-android-clang/clang-r416183b/lib64/clang/12.0.5/lib/linux
+       LIBS += -L/opt/prebuilt-android-clang/clang-r416183b/lib64 -static
+       CC=$(CROSS_COMPILE)gcc
 endif
```

- wlan_src/mapp/uaputl/Makefile

```
--- Makefile    2022-08-29 09:50:15.000000000 -0700
+++ Makefile-uap-update      2022-12-04 18:56:28.747588578 -0800
@@ -23,7 +23,11 @@
 CFLAGS += -Wall
 #ECHO = @
 ifeq (,$(findstring ANDROID_KERNEL, $(CFLAGS)))
-LIBS=-lrt
+       LIBS = -lrt
+else
+       LIBS += -L/opt/prebuilt-android-clang/clang-r416183b/lib64/clang/12.0.5/lib/linux
+       LIBS += -L/opt/prebuilt-android-clang/clang-r416183b/lib64 -static
+       CC=$(CROSS_COMPILE)gcc
 endif
```

- wlan_src/mapp/mlanutl/Makefile

```
--- Makefile    2022-08-29 09:50:15.000000000 -0700
+++ Makefile-mlanutl-update2022-12-04 18:57:31.675862458 -0800
@@ -24,7 +24,11 @@
 CFLAGS += -Wno-stringop-truncation
 #ECHO = @
 ifeq (,$(findstring ANDROID_KERNEL, $(CFLAGS)))
-LIBS=-lrt
+       LIBS = -lrt
+else
+       LIBS += -L/opt/prebuilt-android-clang/clang-r416183b/lib64/clang/12.0.5/lib/linux
+       LIBS += -L/opt/prebuilt-android-clang/clang-r416183b/lib64 -static
+       CC=$(CROSS_COMPILE)gcc
 endif

 .PHONY: default tags all
```

- wlan_src/mapp/mlanevent/Makefile

```
--- Makefile    2022-08-29 09:50:15.000000000 -0700
+++ Makefile-mlanevent-update    2022-12-04 18:58:30.108116769 -0800
@@ -23,7 +23,11 @@
 CFLAGS += -Wall
 #ECHO = @
 ifeq (,$(findstring ANDROID_KERNEL, $(CFLAGS)))
-LIBS=-lrt
+    LIBS = -lrt
+else
+    LIBS += -L/opt/prebuilt-android-clang/clang-r416183b/lib64/clang/12.0.5/lib/linux
+    LIBS += -L/opt/prebuilt-android-clang/clang-r416183b/lib64 -static
+    CC=$(CROSS_COMPILE)gcc
 endif
```

- wlan_src/mapp/mlanconfig/Makefile

```
--- Makefile    2022-08-29 09:50:14.000000000 -0700
+++ Makefile-mlanconfig-update  2022-12-04 18:59:42.520431920 -0800
@@ -19,6 +19,7 @@
 # remove KERNEL include dir
 CFLAGS := $(filter-out -I$(KERNELDIR)%, $(CFLAGS))

+CC=$(CROSS_COMPILE)gcc
 #
 # List of application executables to create
 #
```

- wlan_src/mapp/mlan2040coex/Makefile

```
--- Makefile    2022-08-29 09:50:14.000000000 -0700
+++ Makefile-mlan2040coex-update    2022-12-04 19:00:43.008695165 -0800
@@ -19,6 +19,7 @@
 # remove KERNEL include dir
 CFLAGS := $(filter-out -I$(KERNELDIR)%, $(CFLAGS))

+CC=$(CROSS_COMPILE)gcc
 #
 # List of application executables to create
 #
```

[Note]

For other WiFi drivers, users can tune Makefiles manually according to above updates.

(3) Starting Cross Compilation

- **Setting environment for cross compilation**

  Go back to **wlan_src** directory and set environment cross compiling wifi driver.

```
# export KERNELDIR=/home/weidong/android_11.0.0_2.6.0/out/target/product/mek_8q/obj/KERNEL_OBJ
# export ARCH=arm64
# export PATH=/opt/prebuilt-android-clang/clang-r416183b/bin:${PATH}
# export PATH=/opt/gcc-arm-8.3-2019.03-x86_64-aarch64-linux-gnu/bin:${PATH}
```

```
# export CLANG_CC=clang
# export CLANG_CXX=clang++
# export LLVM_LD=ld.lld
# export CROSS_COMPILE=aarch64-linux-gnu-
```

- **Cross Compilation**

```
# make -C `pwd` LLVM=1 LLVM_IAS=1 build
```

(4) Checking cross compilation results

After cross compilation, the compiled files are all in bin_ wlan directory. bin_ wlan and wlan_src is at the same level.

```
16344.P3-MGPL/WIFI-BT-U16-X86--MM5X17344.p3-MGPL/wlan_src$ ls ../bin_wlan/
config  mlan2040coex   mlan.ko  moal.ko  README_MLAN  README_UAP         uaputl.exe  wifidirect
load    mlanevent.exe  mlanutl  README   README_RBC   README_WIFIDIRECT  unload      wifidirectutl
```

Use 'file' command to check above files' type:

```
# cd ../bin_wlan
# file moal.ko
moal.ko: ELF 64-bit LSB relocatable, ARM aarch64, version 1 (SYSV), BuildID[sha1]=af9f6827e21bcb9733db1978ee01effcc6bbb758, with debug_info, not stripped
```

```
# file mlanutl
mlanutl: ELF 64-bit LSB executable, ARM aarch64, version 1 (GNU/Linux), statically linked, for GNU/Linux 3.7.0, with debug_info, not stripped
```

```
# file uaputl.exe
uaputl.exe: ELF 64-bit LSB executable, ARM aarch64, version 1 (GNU/Linux), statically linked, for GNU/Linux 3.7.0, with debug_info, not stripped
```

[Note]

Since the android system uses clang as the compiler, we also use clang to compile the wifi driver.

NXP Global CAS Connectivity Team
Weidong Sun
12/05/2022