SAMA5D27-SOM1-EK1

This is a page about Microchip's Cortex-A5 SMART SAMA5D27-SOM1 based evaluation platform (ATSAMA5D27-SOM1-EK1).

Availability

Boards:
ATSAMA5D27-SOM1-EK1-ND at Digi-Key

Vendor Documentation

- Microchip Documentation: https://www.microchip.com
  - Microchip Linux4SAM: http://www.at91.com/linux4sam/
  - Microchip SAMA5 Cortex-A5-based MPUs: https://www.microchip.com/design-centers/32-bit-mpus/microprocessors/sama5
  - Microchip SAMA5D2 Series: https://www.microchip.com/design-centers/32-bit-mpus/microprocessors/sama5/sama5d2-series
  - Microchip SAMA5D27-SOM1-EK1: https://www.microchip.com/Developmenttools/ProductDetails/atsama5d27-som1-ek1

Basic Requirements

- Running a recent release of Debian, Fedora or Ubuntu; without OS Virtualization Software.
- ARM Cross Compiler – Linaro: https://www.linaro.org
  - Linaro Toolchain Binaries: https://www.linaro.org/downloads/
- Bootloader
  - Das U-Boot – the Universal Boot Loader: http://www.denx.de/wiki/U-Boot
  - Source: https://github.com/u-boot/u-boot/
- Linux Kernel
  - Linus's Mainline tree: https://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git
- ARM based rootfs
  - Debian: https://www.debian.org
  - Ubuntu: https://www.ubuntu.com

ARM Cross Compiler: GCC

This is a pre-built (64bit) version of Linaro GCC that runs on generic linux, sorry (32bit) x86 users, it's time to upgrade...
Download/Extract:

~/

```
wget -c https://releases.linaro.org/components/toolchain/binaries/6.4-2018.05/arm-linux-gnueabihf/gcc-linaro-6.4.1-2018.05-x86_64_arm-linux-gnueabihf.tar.xz
tar xf gcc-linaro-6.4.1-2018.05-x86_64_arm-linux-gnueabihf.tar.xz
export CC=`pwd`/gcc-linaro-6.4.1-2018.05-x86_64_arm-linux-gnueabihf/bin/arm-linux-gnueabihf-
```

Test Cross Compiler:

~/

```
${CC}gcc --version
arm-linux-gnueabihf-gcc (Linaro GCC 6.4-2018.05) 6.4.1 20180425 [linaro-6.4-2018.05 revision 7b15d0869c096fe39603ad63dc19ab7cf035eb70]
Copyright (C) 2017 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

Bootloader: AT91Bootstrap

AT91Bootstrap – Microchip's first step bootloader: http://www.at91.com/linux4sam/bin/view/Linux4SAM/AT91Bootstrap
Source: https://github.com/linux4sam/at91bootstrap
Download:

~/

```
git clone https://github.com/linux4sam/at91bootstrap
cd at91bootstrap/
git checkout v3.8.10 -b tmp
```

Configure and Build:

~/at91bootstrap

```
make ARCH=arm CROSS_COMPILE=${CC} distclean
make ARCH=arm CROSS_COMPILE=${CC} sama5d27_som1_eksd_uboot_defconfig
make ARCH=arm CROSS_COMPILE=${CC}
```

Bootloader: U-Boot

Das U-Boot – the Universal Boot Loader: http://www.denx.de/wiki/U-Boot
eewiki.net patch archive: https://github.com/eewiki/u-boot-patches
Download:

~/

```
git clone https://github.com/u-boot/u-boot
cd u-boot/
git checkout v2019.01 -b tmp
```

Patches:

~/u-boot

```
wget -c https://github.com/eewiki/u-boot-patches/raw/master/v2019.01/0001-ARM-at91-Convert-SPL_GENERATE_ATMEL_PMECC_HEADER-to-.patch
wget -c https://github.com/eewiki/u-boot-patches/raw/master/v2019.01/0001-sama5dX-fixes.patch

patch -p1 < 0001-ARM-at91-Convert-SPL_GENERATE_ATMEL_PMECC_HEADER-to-.patch
patch -p1 < 0001-sama5dX-fixes.patch
```

Configure and Build:

~/u-boot

```
make ARCH=arm CROSS_COMPILE=${CC} distclean
make ARCH=arm CROSS_COMPILE=${CC} sama5d27_som1_ek_mmc_defconfig
make ARCH=arm CROSS_COMPILE=${CC}
```

Linux Kernel

This script will build the kernel, modules, device tree binaries and copy them to the deploy directory.
Download:

~/

```
git clone https://github.com/RobertCNelson/armv7_devel
cd armv7_devel/
```

For v4.14.x-sama5-armv7 (Longterm 4.14.x):

~/armv7_devel/

```
git checkout origin/v4.14.x-sama5-armv7 -b tmp
```

For v4.19.x-sama5-armv7 (Longterm 4.19.x):

~/armv7_devel/

```
git checkout origin/v4.19.x-sama5-armv7 -b tmp
```

Build:

~/armv7_devel/

```
./build_kernel.sh
```

Root File System

Debian 9

User	Password
debian	temppwd
root	root

Download:

~/

```
wget -c https://rcn-ee.com/rootfs/eewiki/minfs/debian-9.6-minimal-armhf-2018-11-26.tar.xz
```

Verify:

~/

```
sha256sum debian-9.6-minimal-armhf-2018-11-26.tar.xz
c9d9b2623131829eeeefa153274219d89887a1984b8dde72cfbfe29f29c7a10d  debian-9.6-minimal-armhf-2018-11-26.tar.xz
```

Extract:

~/

```
tar xf debian-9.6-minimal-armhf-2018-11-26.tar.xz
```

Ubuntu 18.04 LTS

User	Password
ubuntu	temppwd

Download:

~/

```
wget -c https://rcn-ee.com/rootfs/eewiki/minfs/ubuntu-18.04.1-minimal-armhf-2018-11-26.tar.xz
```

Verify:

~/

```
sha256sum ubuntu-18.04.1-minimal-armhf-2018-11-26.tar.xz
d8190bc858bab9db83bec03707b2636dc23646fca2a0420fb49adbedf7d492c5  ubuntu-18.04.1-minimal-armhf-2018-11-26.tar.xz
```

Extract:

~/

```
tar xf ubuntu-18.04.1-minimal-armhf-2018-11-26.tar.xz
```

Setup microSD card

We need to access the External Drive to be utilized by the target device. Run lsblk to help figure out what linux device has been reserved for your External Drive.

Example: for DISK=/dev/sdX

```
lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0 465.8G  0 disk
sda1   8:1    0   512M  0 part /boot/efi
sda2   8:2    0 465.3G  0 part /               <- Development Machine Root Partition
sdb      8:16   1   962M  0 disk                 <- microSD/USB Storage Device
```

```
sdb1   8:17   1   961M  0 part                  <- microSD/USB Storage
Partition
```

Thus you would use:

```
export DISK=/dev/sdb
```

Example: for DISK=/dev/mmcblkX

```
lsblk
NAME      MAJ:MIN   RM   SIZE RO TYPE MOUNTPOINT
sda         8:0      0 465.8G  0 disk
sda1      8:1      0   512M  0 part /boot/efi
sda2      8:2      0 465.3G  0 part /               <- Development
Machine Root Partition
mmcblk0     179:0    0   962M  0 disk                 <- microSD/USB
Storage Device
mmcblk0p1 179:1    0   961M  0 part                 <- microSD/USB
Storage Partition
```

Thus you would use:

```
export DISK=/dev/mmcblk0
```

Erase partition table/labels on microSD card:

```
sudo dd if=/dev/zero of=${DISK} bs=1M count=50
```

Create Partition Layout:
With util-linux v2.26, sfdisk was rewritten and is now based on libfdisk.

```
sudo sfdisk --version
sfdisk from util-linux 2.27.1
```

sfdisk >= 2.26.x

```
sudo sfdisk ${DISK} <<-__EOF__
1M,48M,0xE,*
49M,,,-
__EOF__
```

sfdisk <= 2.25.x

```
sudo sfdisk --unit M ${DISK} <<-__EOF__
1,48,0xE,*
49,,,-
__EOF__
```

Format Partition:

```
for: DISK=/dev/mmcblkX
sudo mkfs.vfat -F 16 -n BOOT ${DISK}p1
sudo mkfs.ext4 -L rootfs ${DISK}p2

for: DISK=/dev/sdX
sudo mkfs.vfat -F 16 -n BOOT ${DISK}1
sudo mkfs.ext4 -L rootfs ${DISK}2
```

Mount Partition:
On most systems these partitions may will be auto-mounted...

```
sudo mkdir -p /media/boot/
sudo mkdir -p /media/rootfs/

for: DISK=/dev/mmcblkX
sudo mount ${DISK}p1 /media/boot/
sudo mount ${DISK}p2 /media/rootfs/

for: DISK=/dev/sdX
sudo mount ${DISK}1 /media/boot/
sudo mount ${DISK}2 /media/rootfs/
```

Install Bootloader

Copy at91bootstrap/u-boot binaries to the boot partition

~/

```
sudo cp -v ./at91bootstrap/binaries/sama5d27_som1_ek-sdcardboot-uboot-
3.8.10.bin /media/boot/BOOT.BIN
sudo cp -v ./u-boot/u-boot.bin /media/boot/
```

Install Kernel and Root File System

To help new users, since the kernel version can change on a daily basis. The kernel building scripts listed on this page will now give you a hint of what kernel version was built.

```
----------------------------
Script Complete
eewiki.net: [user@localhost:~$ export kernel_version=4.X.Y-Z]
----------------------------
```

Copy and paste that "export kernel_version=4.X.Y-Z" exactly as shown in your own build/desktop environment and hit enter to create an environment variable to be used later.

```
export kernel_version=4.X.Y-Z
```

Copy Root File System

~/

```
sudo tar xfvp ./*-*-*-armhf-*/armhf-rootfs-*.tar -C /media/rootfs/
sync
sudo chown root:root /media/rootfs/
sudo chmod 755 /media/rootfs/
```

Copy Kernel Image

Kernel Image:

~/

```
sudo cp -v ./armv7_devel/deploy/${kernel_version}.zImage /media/boot/zImage
```

Copy Kernel Device Tree Binaries

~/

```
sudo mkdir -p /media/boot/dtbs/
sudo tar xfvo ./armv7_devel/deploy/${kernel_version}-dtbs.tar.gz -C /media
/boot/dtbs/
```

Copy Kernel Modules

~/

```
sudo tar xfv ./armv7_devel/deploy/${kernel_version}-modules.tar.gz -C
/media/rootfs/
```

File Systems Table (/etc/fstab)

```
sudo sh -c "echo '/dev/mmcblk0p2  /  auto  errors=remount-ro  0  1' >>
/media/rootfs/etc/fstab"
sudo sh -c "echo '/dev/mmcblk0p1  /boot/uboot  auto  defaults  0  2' >>
/media/rootfs/etc/fstab"
```

Remove microSD/SD card

```
sync
sudo umount /media/boot
sudo umount /media/rootfs
```

Comments

Comments, feedback, and questions can be sent to: eewiki@digikey.com
Please use the Digi-Key's TechForum: TechForum