**MAKE A DISTRO FOR SAMA5d27 using Ubuntu**

**GENERAL INFO**

if we need to change setting in a file, we find file or string in file by using the following

--------------------------------------------------------------------------------------------------------------------------------

locate \*.config for file or locate “Kconfig” for list

--------------------------------------------------------------------------------------------------------------------------------

or

--------------------------------------------------------------------------------------------------------------------------------

grep --include=\\*.{c,h} --exclude=\*.o -rnw '/path/to/somewere/' -e 'pattern'

--------------------------------------------------------------------------------------------------------------------------------

or

--------------------------------------------------------------------------------------------------------------------------------

grep -Ril "the text to find" /

-------------------------------------------------------------------------------------------------------------------------------

i ignore case

R recursive

l show filename not the result

/ start at the root of machine

--------------------------------------------------------------------------------------------------------------------------------

or to find file (not text inside)

--------------------------------------------------------------------------------------------------------------------------------

find /etc -type f -name "\*.conf"

--------------------------------------------------------------------------------------------------------------------------------

or find lets say anything in kernel that has to do with uarts , tty etc and default =4 do the following

--------------------------------------------------------------------------------------------------------------------------------

Grep –exclude\\*{0,h,c} -rnw ‘.armv7\_devel’ -e ‘default “4”’

**FIRST STEP**

linux Hyper-V connect if you use hyper-v (windows 10 pro) to run linux

Open a linux console (in all cases)

**DO NOT CLOSE CONSOLE IN ANY STEP.... IF YOU DO YOU HAVE TO DO IT ALL FROM START**

--------------------------------------------------------------------------------------------------------------------------------

**INSTALL THIS TO BE ABLE TO GENERATE ISO**

--------------------------------------------------------------------------------------------------------------------------------

***Install once only not every time …***

sudo apt-get install squashfs-tools genisoimage

--------------------------------------------------------------------------------------------------------------------------------

**DO NOT CLOSE CONSOLE IN ANY STEP.... IF YOU DO YPU HAVE TO DO IT ALL FROM START**

**MAIN DISTRO CREATION STARTS HERE**

--------------------------------------------------------------------------------------------------------------------------------

--------------------------------------------------------------------------------------------------------------------------------

ARM Cross Compiler: GCC - This is a pre-built (64bit) version of Linaro GCC that runs on generic linux,

it's time to upgrade...

*if you do this step for the second time and you have already downloaded, the use only the export and {CC}gcc --version*

--------------------------------------------------------------------------------------------------------------------------------

wget -c https://releases.linaro.org/components/toolchain/binaries/6.5-2018.12/arm-linux-gnueabihf/gcc-linaro-6.5.0-2018.12-x86\_64\_arm-linux-gnueabihf.tar.xz

tar xf gcc-linaro-6.5.0-2018.12-x86\_64\_arm-linux-gnueabihf.tar.xz

**DO NOT FORGET THE FOLLOWING**

export CC=`pwd`/gcc-linaro-6.5.0-2018.12-x86\_64\_arm-linux-gnueabihf/bin/arm-linux-gnueabihf-

**check the version**

${CC}gcc --version

--------------------------------------------------------------------------------------------------------------------------------

**BELLOW IF ANY ERRORS FROM MISSING SOFTWARE INSTALL WITH APT the software that the terminal says**

--------------------------------------------------------------------------------------------------------------------------------

--------------------------------------------------------------------------------------------------------------------------------

**Bootloader: AT91Bootstrap**

if you do this step for the second time and you have already downloaded, the use only the make commands

--------------------------------------------------------------------------------------------------------------------------------

git clone https://github.com/linux4sam/at91bootstrap

cd at91bootstrap/

git checkout v3.8.10 -b tmp

--------------------------------------------------------------------------------------------------------------------------------

change the config files variables for tty (Kconfig files in drivers,arch,etc) or anything else and then continue below

--------------------------------------------------------------------------------------------------------------------------------

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**

if you do this step for the second time and you have already downloaded, the use only the cd u-boot and patch and make commands

--------------------------------------------------------------------------------------------------------------------------------

git clone https://github.com/u-boot/u-boot

cd u-boot/

git checkout v2019.01 -b tmp

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

--------------------------------------------------------------------------------------------------------------------------------

change the config files variables for tty or anything else (Kconfig files in drivers,arch,etc) and then continue below

--------------------------------------------------------------------------------------------------------------------------------

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**

if you do this step for the second time and you have already downloaded, the use only the change the config files bellow

--------------------------------------------------------------------------------------------------------------------------------

git clone https://github.com/RobertCNelson/armv7\_devel

cd armv7\_devel/

git checkout origin/v4.14.x-sama5-armv7 -b tmp OR git checkout origin/v4.19.x-sama5-armv7 -b tmp

--------------------------------------------------------------------------------------------------------------------------------

change the config files variables for tty or anything else (Kconfig files in drivers,arch,etc) , before the following build....

--------------------------------------------------------------------------------------------------------------------------------

./build\_kernel.sh

--------------------------------------------------------------------------------------------------------------------------------

**the kernel config will start to check what we need from the kernel like SPI etc etc**

--------------------------------------------------------------------------------------------------------------------------------

-------------------------------------------------------------------------------------------------------------------------------

**Use Debian or Ubuntu (one of the two)**

if you do this step for the second time and you have already downloaded, then do not use this step

--------------------------------------------------------------------------------------------------------------------------------

***Debian 10***

User Pwd

debian temppwd

root root

--------------------------------------------------------------------------------------------------------------------------------

wget -c https://rcn-ee.com/rootfs/eewiki/minfs/debian-10.1-minimal-armhf-2019-09-16.tar.xz

sha256sum debian-10.1-minimal-armhf-2019-09-16.tar.xz

tar xf debian-10.1-minimal-armhf-2019-09-16.tar.xz

--------------------------------------------------------------------------------------------------------------------------------

***Ubuntu 18.04 LTS***

user pwd

ubuntu temppwd

--------------------------------------------------------------------------------------------------------------------------------

wget -c https://rcn-ee.com/rootfs/eewiki/minfs/ubuntu-18.04.3-minimal-armhf-2019-09-16.tar.xz

sha256sum ubuntu-18.04.3-minimal-armhf-2019-09-16.tar.xz

tar xf ubuntu-18.04.3-minimal-armhf-2019-09-16.tar.xz

--------------------------------------------------------------------------------------------------------------------------------

change the config files variables for tty or anything else and then continue below

--------------------------------------------------------------------------------------------------------------------------------

--------------------------------------------------------------------------------------------------------------------------------

**Setup microSD card or new hyper-v disk (2gb set it on settings)**

--------------------------------------------------------------------------------------------------------------------------------

Lsblk

export DISK=/dev/sdb or export DISK=/dev/mmcblk0

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

sudo sfdisk --version

--------------------------------------------------------------------------------------------------------------------------------

**for : sfdisk >= 2.26.x (copy the following 4 lines all together to terminal)**

--------------------------------------------------------------------------------------------------------------------------------

sudo sfdisk ${DISK} <<-\_\_EOF\_\_

1M,48M,0xE,\*

49M,,,-

\_\_EOF\_\_

--------------------------------------------------------------------------------------------------------------------------------

**for : sfdisk <= 2.25.x (copy the following 4 lines all together to terminal)**

--------------------------------------------------------------------------------------------------------------------------------

sudo sfdisk --unit M ${DISK} <<-\_\_EOF\_\_

1,48,0xE,\*

49,,,-

\_\_EOF\_\_

--------------------------------------------------------------------------------------------------------------------------------

**Format Partitions: find from the lsblk**

--------------------------------------------------------------------------------------------------------------------------------

--------------------------------------------------------------------------------------------------------------------------------

**for: DISK=/dev/mmcblkX**

--------------------------------------------------------------------------------------------------------------------------------

sudo mkfs.vfat -F 16 -n BOOT ${DISK}p1

sudo mkfs.ext4 -L rootfs ${DISK}p2

--------------------------------------------------------------------------------------------------------------------------------

**for: DISK=/dev/sdX wrere X find from the lsblk**

--------------------------------------------------------------------------------------------------------------------------------

sudo mkfs.vfat -F 16 -n BOOT ${DISK}1

sudo mkfs.ext4 -L rootfs ${DISK}2

--------------------------------------------------------------------------------------------------------------------------------

**Mount Partition:**

--------------------------------------------------------------------------------------------------------------------------------

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**

--------------------------------------------------------------------------------------------------------------------------------

~/

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**

--------------------------------------------------------------------------------------------------------------------------------

***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 change the values 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**

--------------------------------------------------------------------------------------------------------------------------------

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"

--------------------------------------------------------------------------------------------------------------------------------

**OR if sbdX where X find from the lsblk command**

--------------------------------------------------------------------------------------------------------------------------------

sudo sh -c "echo '/dev/sbdp2 / auto errors=remount-ro 0 1' >> /media/rootfs/etc/fstab"

sudo sh -c "echo '/dev/sbdp1 /boot/uboot auto defaults 0 2' >> /media/rootfs/etc/fstab"

--------------------------------------------------------------------------------------------------------------------------------

**Remove microSD/SD card**

--------------------------------------------------------------------------------------------------------------------------------

sync

sudo umount /media/boot

sudo umount /media/rootfs

--------------------------------------------------------------------------------------------------------------------------------

**CREATE a .bin DISK IMAGE dd if=<device> of=<image>**

--------------------------------------------------------------------------------------------------------------------------------

sudo fdisk -l

sudo dd if=/dev/sdb of=./imagename.img

--------------------------------------------------------------------------------------------------------------------------------

on linux (original not the build)

mount the sdbX

--------------------------------------------------------------------------------------------------------------------------------

udisksctl mount -b /dev/sdbX where X the 1 or 2 (boot partition or space)

go to folder and copy anything else you might need

if you need to get something from another computer either use smb on a shared drive

or on the other computer you must have a web server so you do the following

wget http://ip\_of\_computer/directory\_of\_file/file

--------------------------------------------------------------------------------------------------------------------------------

**CREATE ISO FROM GENERATED IMAGE**

--------------------------------------------------------------------------------------------------------------------------------

sudo genisoimage -D -r -V "$IMAGE\_NAME" -cache-inodes -J -l -b isolinux/isolinux.bin -c isolinux/boot.cat -no-emul-boot -boot-load-size 4 -boot-info-table -o ../name-of-your-custom-image.iso .

--------------------------------------------------------------------------------------------------------------------------------

AFTER BOOT on the new image booted on the arm

--------------------------------------------------------------------------------------------------------------------------------

--------------------------------------------------------------------------------------------------------------------------------

**check tty**

--------------------------------------------------------------------------------------------------------------------------------

dmesg | grep tty

--------------------------------------------------------------------------------------------------------------------------------

**check space**

--------------------------------------------------------------------------------------------------------------------------------

df

--------------------------------------------------------------------------------------------------------------------------------

**Check memory**

--------------------------------------------------------------------------------------------------------------------------------

free -m

vmstat -s

top

cat /proc/meminfo

--------------------------------------------------------------------------------------------------------------------------------

**Hardware info on memory**

--------------------------------------------------------------------------------------------------------------------------------

sudo dmidecode -t 17

--------------------------------------------------------------------------------------------------------------------------------

on the new distro after boot if we need to run a file that we have created ,

first get the file in some folder and then

--------------------------------------------------------------------------------------------------------------------------------

chmod +x filename

file filename

./filename

--------------------------------------------------------------------------------------------------------------------------------

file and chmode only the first time

file shows statistics on the executable (like processor etc) and chmod +x add exec right to file

--------------------------------------------------------------------------------------------------------------------------------

--------------------------------------------------------------------------------------------------------------------------------

**CUSTOMIZE (ADD OR DELETE SOFTWARE) IN INSTALL DISK**

-------------------------------------------------------------------------------------------------------------------------------

First download the ISO you want to start customizing from Ubuntu’s release server. Remember where you save it,

because you’re going to have to move it in a minute OR use the above created img distro

From here on out, it’s bash command line.

Make sure the prerequisite software is installed for unpacking and repacking the image. Otherwise Open a terminal and run:

sudo apt-get install squashfs-tools genisoimage

Create a fresh folder to begin work. For the purposes of this guide, everything will be done from the starting point

of the user’s home directory (indicated in Linux by a tilde “~”). Approximately 10 gigabytes total of free hard

drive space is required for decompressing the ISO filesystem and repackaging it at the end.

mkdir ~/custom-img

Move the base ISO downloaded in the first step to the working directory. From here on out, replace “ubuntu.iso” with the

name of the image downloaded from the Ubuntu Release server ex. trusty-desktop-amd64.iso

cp /path/to/saved/ubuntu.iso ~/custom-img

cd ~/custom-img

Next, extract the contents of disc image.

mkdir mnt

sudo mount -o loop ubuntu.iso mnt

mkdir extract

sudo rsync --exclude=/casper/filesystem.squashfs -a mnt/ extract

Here’s where things start to get interesting. Extract the filesystem with the following commands:

sudo unsquashfs mnt/casper/filesystem.squashfs

sudo mv squashfs-root edit

You’re going to need network access from within the chroot environment to download and install updated/new packages.

Essentially what’s happening is you are going to “log in” to a command line instance of the Ubuntu installation,

separate from the host system. Copy resolv.conf from your system into the freshly unpacked fs.

sudo cp /etc/resolv.conf edit/etc/

Mount a few important working directories:

sudo mount --bind /dev/ edit/dev

sudo chroot edit

mount -t proc none /proc

mount -t sysfs none /sys

mount -t devpts none /dev/pts

Now you are actually logged in to the installation instance as root. Before making changes, a few

commands will make sure that everything goes smoothly while modifying packages.

export HOME=/root

export LC\_ALL=C

dbus-uuidgen > /var/lib/dbus/machine-id

dpkg-divert --local --rename --add /sbin/initctl

ln -s /bin/true /sbin/initctl

Start by removing the packages you don’t want. Be sure to use the “purge” command so that the system will

automatically uninstall and delete the package, which optimizes the space required for the ISO.

When you execute purge, read the list of programs to be removed before you select “Y” and make

absolutely sure you haven’t accidentally flagged a core system package via association.

You will recognize this because the list will contain significantly more packages than those you selected.

apt-get purge package1 package2 package3

I personally remove games, scanning utilities and default text editors like abiword and mousepad. Stay away from core components unless you know what you are doing.

Since I am customizing a 64-bit Ubuntu image, I need multiarch (i386) support for some of the

programming libraries. The following command is not necessary for everyone, but I recommend it anyway.

dpkg --add-architecture i386

Update the software repositories and upgrade the remaining packages on the system.

apt-get update && apt-get upgrade

Add packages to the system the usual way:

apt-get install package1 package2 package3

Time to clean up:

apt-get autoremove && apt-get autoclean

rm -rf /tmp/\* ~/.bash\_history

rm /var/lib/dbus/machine-id

rm /sbin/initctl

dpkg-divert --rename --remove /sbin/initctl

Unmount the directories from the beginning of this guide:

umount /proc || umount -lf /proc

umount /sys

umount /dev/pts

exit

sudo umount edit/dev

You have now “logged out” of the installation environment and are “back” on the host system.

These final steps will actually produce the ISO.

Generate a new file manifest:

sudo chmod +w extract/casper/filesystem.manifest

sudo chroot edit dpkg-query -W --showformat='${Package} ${Version}\n' | sudo tee extract/casper/filesystem.manifest

sudo cp extract/casper/filesystem.manifest extract/casper/filesystem.manifest-desktop

sudo sed -i '/ubiquity/d' extract/casper/filesystem.manifest-desktop

sudo sed -i '/casper/d' extract/casper/filesystem.manifest-desktop

Compress the filesystem:

sudo mksquashfs edit extract/casper/filesystem.squashfs -b 1048576

Update filesystem size (needed by the installer):

printf $(sudo du -sx --block-size=1 edit | cut -f1) | sudo tee extract/casper/filesystem.size

Delete the old md5sum:

cd extract

sudo rm md5sum.txt

…and generate a fresh one: (single command, copy and paste in one piece)

find -type f -print0 | sudo xargs -0 md5sum | grep -v isolinux/boot.cat | sudo tee md5sum.txt

And finally, create the ISO. This is a single long command, be sure to copy and paste it in one piece

and don’t forget the period at the end, it’s important:

sudo genisoimage -D -r -V "$IMAGE\_NAME" -cache-inodes -J -l -b isolinux/isolinux.bin -c isolinux/boot.cat -no-emul-boot -boot-load-size 4 -boot-info-table -o ../name-of-your-custom-image.iso .

It takes a few minutes, but when that is done you will have a burnable/distributable ISO in your

working directory (~/custom-img)