■気づいたら

12 月.
なんだか暖かいままだし，10 月くらいの気分 ...
とかいってるとやばい.

■ ZCU111 の復習

2020.2 版の TRD がでてたので動作確認がてら
PetaLinux を使った ZCU111 向けのビルド手順の再確認をした

```
unzip rdf0476-zcu111-rf-dc-eval-tool-2020-2.zip
cd rdf0476-zcu111-rf-dc-eval-tool-2020-2/apu/
petalinux-create -t project -s rfsoc_petalinux_bsp.bsp
cd rfsoc_petalinux_bsp/
petalinux-config --get-hw-description=../../pl/MTSDesign_8x8/pre-built/rfsoc_trd
petalinux-build
petalinux-package --force --boot --fsbl zynqmp_fsbl.elf --pmufw pmufw.elf --u-boot u-boot.elf
ls -l BOOT.BIN image.ub boot.scr
```

BOOT.BIN image.ub boot.scr を SD カードにコピー.
autostart.sh の中身は

```
#!/bin/sh
ifconfig -a | grep eth0
RESULT=$?
if [ $RESULT -eq 0 ]; then
        ifconfig eth0 192.168.1.3
        rftool
fi
```

なので，そのまま流用.

■ PetaLinux を使わずに Zynq MPSoC 向けに Linux などをビルド

2019.1 で作業する

必要なソースコードをダウンロード

https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842156/Fetch+Sources

```
$ cd ${WORK}
$ git clone https://github.com/Xilinx/linux-xlnx.git
$ git clone https://github.com/Xilinx/u-boot-xlnx.git
$ git clone https://github.com/Xilinx/device-tree-xlnx.git
$ git clone https://git.kernel.org/pub/scm/utils/dtc/dtc.git
$ git clone https://github.com/Xilinx/arm-trusted-firmware.git
$ git clone https://github.com/Xilinx/xen.git
$ git clone https://github.com/Xilinx/embeddedsw.git
```

FSBL のビルド

https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18841798/Build+FSBL

適当に作った hdf をコピー.

```
$ export CROSS_COMPILE=aarch64-linux-gnu-
$ mkdir ${WORK}/fsbl; cd ${WORK}/fsbl
```

```
$ cp ${WORK}/zcu111_base/zcu111_base.sdk/desgin_1_wrapper.hdf .
```

で,

```
$ hsi
hsi% set hwdsgn [open_hw_design design_1_wrapper.hdf]
hsi% generate_app -hw $hwdsgn -os standalone -proc psu_cortexa53_0 -app zynqmp_fsbl -compile -sw fsbl -dir fsbl
```

dtc のビルド

https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18841988/Build+Device+Tree+Compiler+dtc

これはホスト上のプログラムのビルド (のはず)

```
$ sudo apt install pkg-config libyaml-dev # if required
$ cd ${WORK}/dtc
$ make
```

PMU ファームウェアのビルド

https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842462/Build+PMU+Firmware

```
$ export CROSS_COMPILE=aarch64-linux-gnu-
$ mkdir ${WORK}/pmc; cd ${WORK}/pmc
$ cp ${WORK}/zcu111_base/zcu111_base.sdk/desgin_1_wrapper.hdf .
$ hsi
hsi% set hwdsgn [open_hw_design design_1_wrapper.hdf]
hsi% generate_app -hw $hwdsgn -os standalone -proc psu_pmu_0 -app zynqmp_pmufw -compile -sw pmufw -dir pmu
```

ARM Trusted Firmware のビルド

https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842305/Build+ARM+Trusted+Firmware+ATF

```
$ export CROSS_COMPILE=aarch64-linux-gnu-
$ cd ${WORK}/arm-trusted-firmware
$ make PLAT=zynqmp RESET_TO_BL31=1
```

./build/zynqmp/release/bl31/bl31.elf ができる

U-BOOT

https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18841973/Build+U-Boot

```
$ export CROSS_COMPILE=aarch64-linux-gnu-
$ export ARCH=aarch64
$ cd ${WORK}/u-boot-xlnx
$ make distclean
$ make xilinx_zynqmp_virt_defconfig
$ export DEVICE_TREE="zynqmp-zcu111-revA"
$ make
```

u-boot や tools/mkimage などができる

RootFS

https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842473/Build+and+Modify+a+Rootfs

```
$ cd ${WORK}
 $  wget  -O  ramdisk.image.gz  "https://xilinx-wiki.atlassian.net/wiki/download/attachments/18842473
/arm_ramdisk.image.gz?version=1&modificationDate=1536675884034&cacheVersion=1&api=v2"
$ mkimage -A arm64 -T ramdisk -C gzip -d ramdisk.image.gz uramdisk.image.gz
```

Device tree

https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842279/Build+Device+Tree+Blob
Xilinx の DTG を使ってハードウェア定義情報から Device Tree ファイルを作る

```
$ cd ${WORK}/device-tree-xlnx
$ git checkout xilinx-v2019.1
$ cp zcu111_base/zcu111_base.sdk/design_1_wrapper.hdf .
$ hsi
hsi% open_hw_design design_1_wrapper.hdf
hsi% set_repo_path .
hsi% create_sw_design device-tree -os device_tree -proc psu_cortexa53_0
hsi% generate_target -dir my_dts
hsi% close_hw_design [current_hw_design]
hsi% exit
```

my_dts/ の下にいろいろファイルが作られたので，dtb を作る

```
$ cd my_dts
$ gcc -I my_dts -E -nostdinc -undef -D__DTS__ -x assembler-with-cpp -o system.dts system-top.dts
$ dtc -I dts -O dtb -o system.dtb system.dts
```

カーネルのビルド

https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842481/Build+kernel

```
$ export CROSS_COMPILE=aarch64-linux-gnu-
$ cd ${WORK}/linux-xlnx
$ make ARCH=arm64 xilinx_zynqmp_defconfig
$ make ARCH=arm64 menuconfig
$ make ARCH=arm64
```

boot.bin の生成

https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18841976/Prepare+boot+image

```
$ cd ${WORK}
```

次の内容で boot.bif を作る

```
image : {
        [destination_cpu = a53-0, bootloader]fsbl.elf
```

```
        [pmufw_image]pmufw.elf
        [destination_cpu = a53-0, exception_level = el-3, trustzone]bl31.elf
        [destination_cpu = a53-0, exception_level = el-2]u-boot.elf
}
```

で，必要なファイルを集めて bootgen

```
$ cp fsbl/fsbl/executable.elf fsbl.elf
$ cp pmc/pmu/executable.elf pmufw.elf
$ cp arm-trusted-firmware/build/zynqmp/release/bl31/bl31.elf .
$ cp u-boot-xlnx/u-boot.elf .
$ bootgen -image boot.bif -arch zynqmp -w -o i BOOT.bin
```

SD カードの用意

https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18841655/Prepare+Boot+Medium

先頭に 200M くらいの FAT 領域，残りに ext4 領域を作る
FAT 領域のタイプは c(W95 FAT32 (LBA))，ext4 の方は 83(Linux) にセットする．
で，

```
mkfs.vfat -F 32 -n boot /dev/sdX1
mkfs.ext4 -L root /dev/sdX2
```

などとしてフォーマット．X のとこは自分の環境にあわせる．

コピー

あとは，SD カードの先頭に作った FAT パーティションに

- BOOT.bin
- linux-xlnx/arch/arm64/boot/Image
- linux-xlnx/arch/arm64/boot/dts/xilinx/zynqmp-zcu111-revA.dtb
- device-tree-xlnx/my_dts/system.dtb
- uramdisk.image.gz

を SD カードにコピー．

起動してみる

途中でインタラプトして，

```
ZynqMP> fatload mmc 0 0x03000000 Image
ZynqMP> fatload mmc 0 0x02a00000 system.dtb
ZynqMP> fatload mmc 0 0x20000000 uramdisk.image.gz
ZynqMP> setenv bootargs earlycon clk_ignore_unused root=/dev/ram rw earlyprintk
ZynqMP> booti 0x03000000 0x20000000 0x02a00000
```

とすれば起動する．

自分で作った dtb だと SD カードを認識しなかった /dev/mmcblk0 が生えなかったけど，

```
ZynqMP> fatload mmc 0 0x03000000 Image
ZynqMP> fatload mmc 0 0x02a00000 zynqmp-zcu111-revA.dtb
ZynqMP> fatload mmc 0 0x20000000 uramdisk.image.gz
ZynqMP> setenv bootargs earlycon clk_ignore_unused root=/dev/ram rw earlyprintk
ZynqMP> booti 0x03000000 0x20000000 0x02a00000
```

としたら /dev/mmcblk0 が生えた

というわけで,

```
ZynqMP> fatload mmc 0 0x03000000 Image
ZynqMP> fatload mmc 0 0x02a00000 zynqmp-zcu111-revA.dtb
 ZynqMP>  setenv  bootargs  earlycon  clk_ignore_unused  consoleblank=0  root=/dev/mmcblk0 p2
rootfstype=ext4 rw rootwait
ZynqMP> booti 0x03000000 - 0x02a00000
```

として，第二パーティションから起動.

Ubuntu をセットアップする

ZYBO で Ubuntu を動かしたとき
https://github.com/fpga-design-contest/ad-refkit/blob/master/docs/sec4/index.md
と同じようにセットアップ

```
$ sudo apt install debootstrap # if required
$ mkdir -p arm64
$ sudo debootstrap --foreign --arch arm64 bionic ./arm64 http://ports.ubuntu.com/
$ sudo apt install qemu-user-static
$ sudo cp /usr/bin/qemu-aarch64-static ./arm64/usr/bin/
$ sudo chroot ./arm64
I have no name!@qdev:/# ./debootstrap/debootstrap --second-stage
I have no name!@qdev:/# passwd
I have no name!@qdev:/# su
root@ubuntu:# passwd
root@ubuntu:# locale-gen ja_JP.UTF-8 en_US.UTF-8
root@ubuntu:# apt install software-properties-common
root@ubuntu:# add-apt-repository universe && apt update
root@ubuntu:# apt install build-essential flex bison net-tools
root@ubuntu:# apt install openssh-server
root@ubuntu:# systemctl enable ssh
root@ubuntu:# adduser user
root@ubuntu:# apt install git
root@ubuntu:# mkdir -p /usr/local/src/; cd /usr/local/src
root@ubuntu:# git clone -b v1.4.7 https://git.kernel.org/pub/scm/utils/dtc/dtc.git dtc && cd dtc
root@ubuntu:# make && make HOME=/usr install-bin
root@ubuntu:# cd .. && rm -rf dtc
root@ubuntu:# sed -i -e 's/#PasswordAuthentication/PasswordAuthentication/g' /etc/ssh/sshd_config
 root@ubuntu:#  sed  -i  -e  's/#PermitRootLogin  prohibit-password/PermitRootLogin  yes/g'
/etc/ssh/sshd_config
root@ubuntu:# sed -i -e 's/AcceptEnv/#AcceptEnv/g' /etc/ssh/sshd_config
```

■結婚記念日

でした．16 年目だそうです．妻に感謝.