The following shows how to use SDTGen to create a System Device Tree from a XSA file.

• XSCT Setup:
-------------
        Get the path of XSCT binary from the installed Vitis tool
        (say: /home/abc/Xilinx/Vitis/2024.2/bin/xsct)

• Create a sdt.tcl file
----------------------
        set outdir [lindex $argv 1]
        set xsa [lindex $argv 0]
        exec rm -rf $outdir
        sdtgen set_dt_param -xsa $xsa -dir $outdir
        sdtgen generate_sdt

• Run the below command to create a System Device Tree from a .xsa file
        <xsct binary path> sdt.tcl <Vivado generated xsa path> <sdt outdir where
files will be generated>
Example:
        /home/abc/Xilinx/Vitis/2024.2/bin/xsct sdt.tcl design1_wrapper.xsa sdt_outdir

• Change the Yocto build directory and run the below command to export
gen-machineconf tool.
        host@host~$ cd <yocto project build directory>
        host@host~/yocto-sdk/build$ export
PATH=$PATH:<ABSOLUTE_PATH>/gen-machine-conf
Example:
        host@host~/yocto-sdk/build$ export
PATH=$PATH:/home/iwave/yocto-sdk/sources/meta-xilinx/gen-machine-conf/

• Run the below command to generate the machine file from the SDT.
        host@host~/yocto-sdk/build$ host@host~/yocto-sdk/build$ gen-machineconf
--soc-family <soc-family> --soc-variant <soc-variant> parse-sdt --hw-description
<path_to_sdtgen_output_directory> -c <conf-directory> -l
<path-to-build-directory>/build/conf/local.conf --machine-name zynqmp-iwg36m
Example:
        host@host~/yocto-sdk/build$ gen-machineconf --soc-family zynqmp --soc-variant
5ev parse-sdt --hw-description /home/iwave/sdt_outdir -c
/home/iwave/yocto-sdk/sources/meta-iwave/conf/ -l /home/iwave/yocto-
sdk/build/conf/local.conf --machine-name zynqmp-iwg36m


For --soc-family and --soc-variant option refer this README file.

<path-to-build-directory>/../sources/meta-xilinx/gen-machine-conf/README.md


########################################
#                                      #

```
#       Yocto configuration             #
#                                       #
#########################################

Changes to be made in the yocto bsp:
-----------------------------------

device-tree layer
=================

The devie tree layer will be located here <DEVICE_TREE_LAYER_ABSOLUTE_PATH>

Example : /home/iwave/sources/meta-iwave/recipes-bsp/device-tree/

1. device-tree.bbappend
        1.remove the do_configure:append function
        2.remove the SYSTEM_USER_DTSI ?= "system-user.dtsi" line.
        3.Give the system-user.dtsi file name here SRC_URI:append = "
file://<dtsi_file_name> "

Note : <dtsi_file_name> file will be in the below path.

Example,
/home/iwave/sources/meta-iwave/recipes-bsp/device-tree/files/system-user.dtsi

2. device-tree-sdt.inc

        Remove this ,
        SRC_URI:append = " file://system-conf.dtsi"
        EXTRA_OVERLAYS:append:linux = " system-user.dtsi"

        Then add this,
        EXTRA_DT_INCLUDE_FILES:append:linux = " system-user.dtsi"
        EXTRA_DT_INCLUDE_FILES:append:linux-gnueabi = " system-user.dtsi"

Now you are done with the device-tree layer. Next, move to the embeddedsw layer.

embeddedsw layer
================

The embeddedsw layer will be located here <FSBL_LAYER_ABSOLUTE_PATH>

Example : /home/iwave/sources/meta-iwave/recipes-bsp/embeddedsw/

1. In fsbl-firmware_%.bbappend, give the proper psu_init.c file paths in the
do_compile:prepend function.

do_compile:prepend(){
        install -m 0644
```

```
${TOPDIR}/tmp/work/<MACHINE-NAME>-xilinx-linux/device-tree/xilinx-v2024.2+git/device-tree-build/device-tree/psu_init.c ${B}/fsbl-firmware/psu_init.c
        grep -L iwg36m_early_init ${B}/fsbl-firmware/psu_init.c | xargs -r sed -i '/psu_peripherals_init_data()/a\\tstatus &= iwg36m_early_init();'

}
```

2. In fsbl.patch :

Example :
/home/iwave/sources/meta-iwave/recipes-bsp/embeddedsw/files/0001-iW-XXX-YY-DD-RZ.Z-RELx.y-SDx.y-FSBL-V24.2.patch

*In the fsbl patch. Change all the XPAR_XIICPS_0_DEVICE_ID to XPAR_XIICPS_0_BASEADDR
*In the fsbl patch. Change all the XPAR_XIICPS_1_DEVICE_ID to XPAR_XIICPS_1_BASEADDR

conf layer:
==========
Note: After we ran gen-machineconf command,

```
host@host~/yocto-sdk/build$ gen-machineconf --soc-family zynqmp --soc-variant 4ev parse-sdt --hw-description /home/iwave/sdt_outdir -c /home/iwave/yocto-sdk/sources/meta-iwave/conf/ -l /home/iwave/yocto-sdk/build/conf/local.conf --machine-name zynqmp-iwg36m
```

The existing machine file will be overwritten.

The rootfs packages will be removed from the machine file. So add the rootfs packages.

Add the rootfs configs in the below file.

/home/iwave/sources/meta-iwave/conf/machine/<MACHINE-NAME>.conf

```
#Rootfs configs
IMAGE_INSTALL:append = " i2c-tools tzdata usbutils util-linux util-linux-fsck util-linux-umount util-linux-mount util-linux-sfdisk util-linux-blkid util-linux-mkfs util-linux-fdisk ethtool util-linux-mkfs.cramfs watchdog openssh-sftp-server pciutils-ids open-amp coreutils perf perf-tests v4l-utils libv4l media-ctl packagegroup-xilinx-audio packagegroup-xilinx-matchbox packagegroup-core-x11 iperf3 e2fsprogs-mke2fs dosfstools pciutils run-postinsts fpga-manager-script packagegroup-core-boot tcf-agent bridge-utils udev-extraconf linux-xlnx-udev-rules libdfx bootscript libmali-xlnx libmali-xlnx-dev tricube nfs-utils-dev can-utils gstreamer-vcu-examples packagegroup-xilinx-gstreamer libdrm gstreamer1.0 libdrm-tests wpa-supplicant iw bluez5 bluez5-obex obexftp linux-firmware-bcm43455 hostapd
```

kernel-module-hdmi libdrm-amdgpu"

- After configuring the yocto files to build the binaries, execute the below command.
  host@host~ /yocto-sdk/build$ bitbake petalinux-image-minimal

Note: Make sure that Vivado version used for design and Vitis tool used for SDTgen should be same.