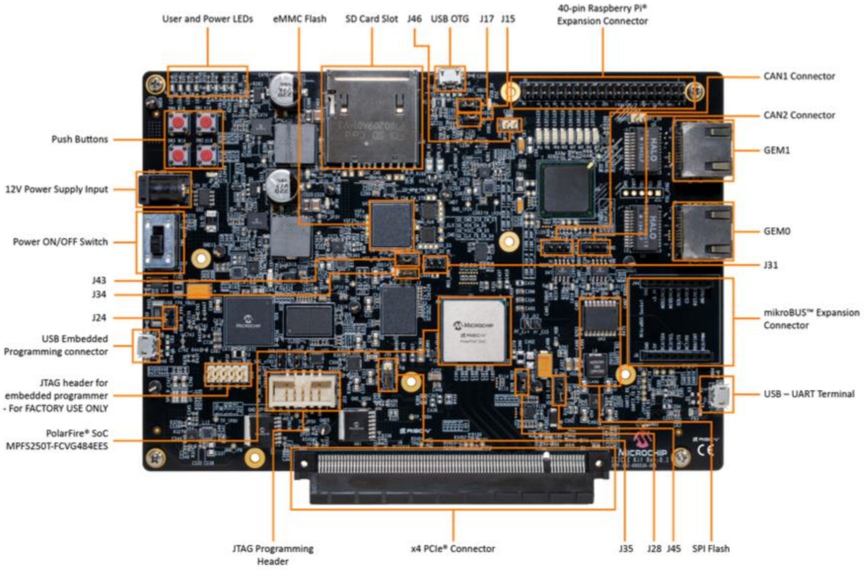
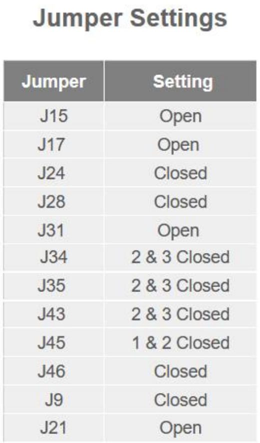
Defect Detection Inference Deployment on PolarFire Icicle Kit

For getting started information on the Icicle Kit please see the PolarFire SoC Icicle Kit quick start guide.

Community support on github and other forums is also helpful.





Jumper Settings

Jumper	Setting
J15	Open
J17	Open
J24	Closed
J28	Closed
J31	Open
J34	2 & 3 Closed
J35	2 & 3 Closed
J43	2 & 3 Closed
J45	1 & 2 Closed
J46	Closed
J9	Closed
J21	Open

Figure 1: Hardware Kit and Jumper setting

Update the Icicle kit to the latest reference design and Hart Software System (HSS)

For updating the latest MSS (Microcontroller Subsystem) Configurator design and HSS (Hart-Software System):

- Download and install the Libero SoC Development Suite from here.
    - To verify the software downloads:
      Enter the following command on terminal to validate the checksum:
      md5sum <path_to_installer>
      sha256sum <path_to_installer>
- Flash a latest Libero MSS component for FPGA design and HSS using Libero SoC. (Link)

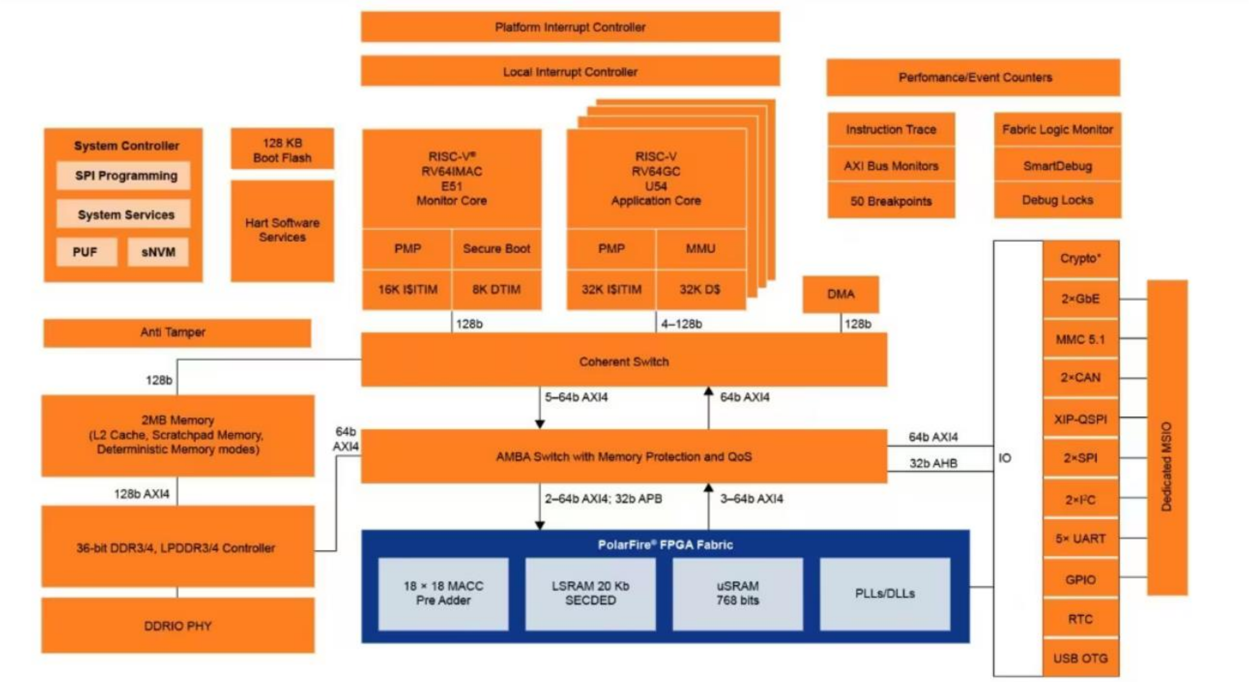
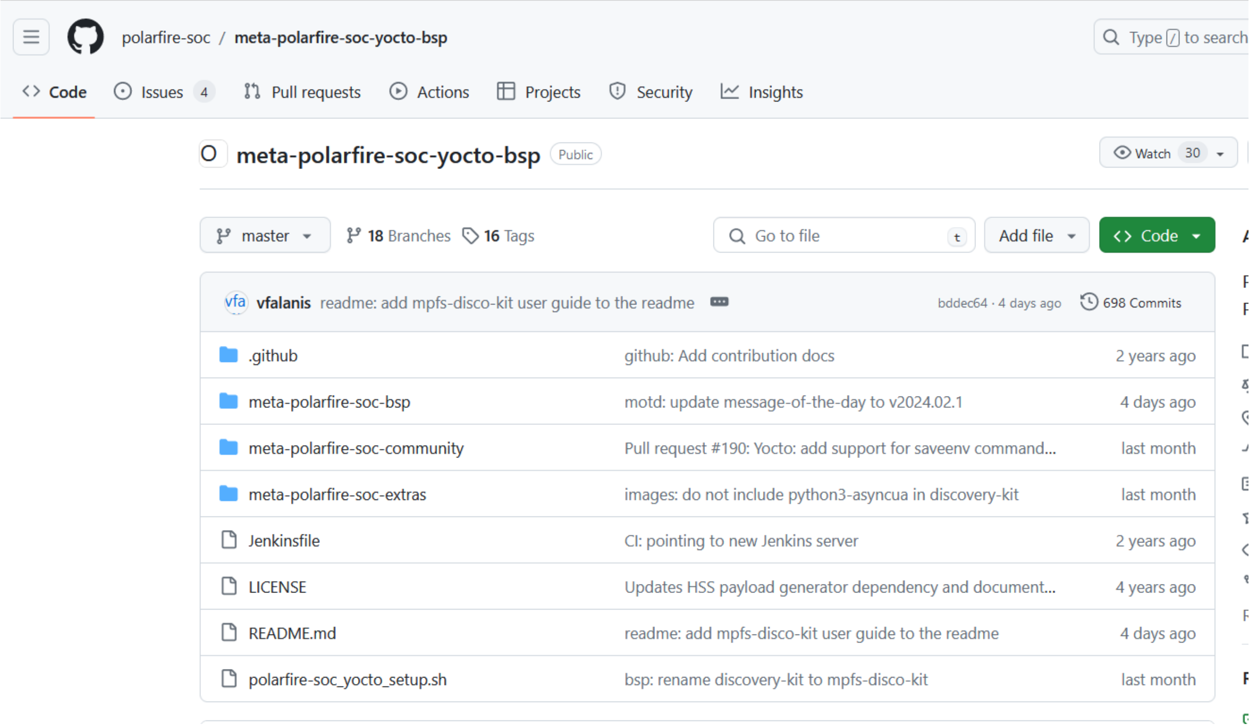


Figure 2: Block diagram: We can enable peripherals as needed.

For Custom Linux:



polarfire-soc / meta-polarfire-soc-yocto-bsp

Code · Issues 4 · Pull requests · Actions · Projects · Security · Insights

meta-polarfire-soc-yocto-bsp Public

Watch 30

master · 18 Branches · 16 Tags · Go to file · Add file · Code

vfalanis readme: add mpfs-disco-kit user guide to the readme · bddec64 · 4 days ago · 698 Commits

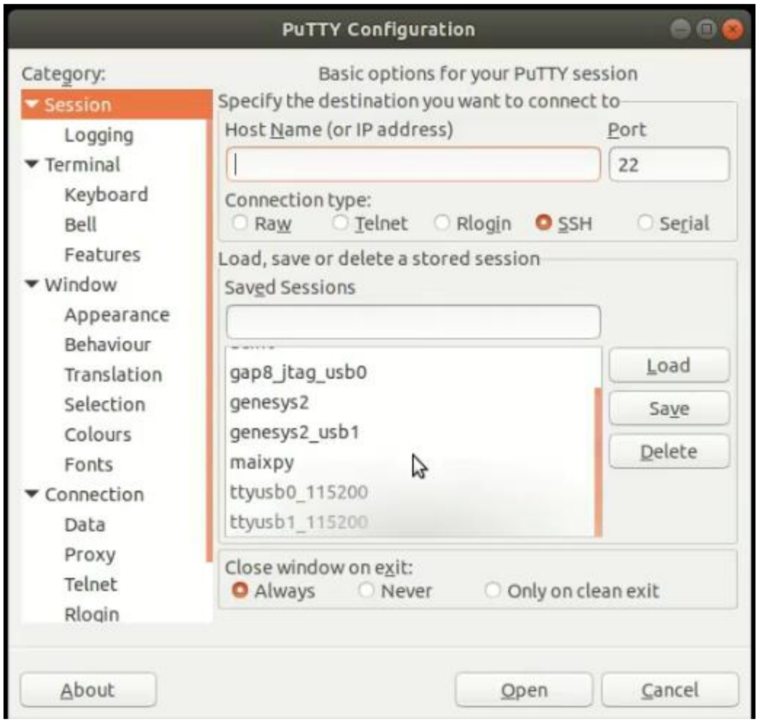
.github	github: Add contribution docs	2 years ago
meta-polarfire-soc-bsp	motd: update message-of-the-day to v2024.02.1	4 days ago
meta-polarfire-soc-community	Pull request #190: Yocto: add support for saveenv command...	last month
meta-polarfire-soc-extras	images: do not include python3-asyncua in discovery-kit	last month
Jenkinsfile	CI: pointing to new Jenkins server	2 years ago
LICENSE	Updates HSS payload generator dependency and document...	4 years ago
README.md	readme: add mpfs-disco-kit user guide to the readme	4 days ago
polarfire-soc_yocto_setup.sh	bsp: rename discovery-kit to mpfs-disco-kit	last month

Figure 3: git-repo for custom Linux using Yocto

- Create the workspace.
  - $ mkdir yocto-dev && cd yocto-dev
  - $ repo init -u https://github.com/polarfire-soc/polarfire-soc-yocto-manifests.git -b main -m default.xml
- Setup BitBake environment.
  - $. ./meta-polarfire-soc-yocto-bsp/polarfire-soc_yocto_setup.sh
- Building a Linux image.
  - MACHINE=icicle-kit-es bitbake mpfs-dev-cli
- Copy the created Disk Image to flash device (USB mmc flash)
  - cd yocto-dev/build
  - bmaptool copy tmp-glibc/deploy/images/icicle-kit-es/mpfs-dev-cli-icicle-kit-es.wic /dev/sdX
- Target machine Linux login.
  - Login with `root` account, there is no password set.

Download Putty: Serial-port terminal

- Open two Putty terminal.
- For the configuration:
  - Connection - Serial
  - Device name - /dev/ttyUSB0 and /dev/ttyUSB1
  - Baud rate - 115200



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address):

Port: 22

Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

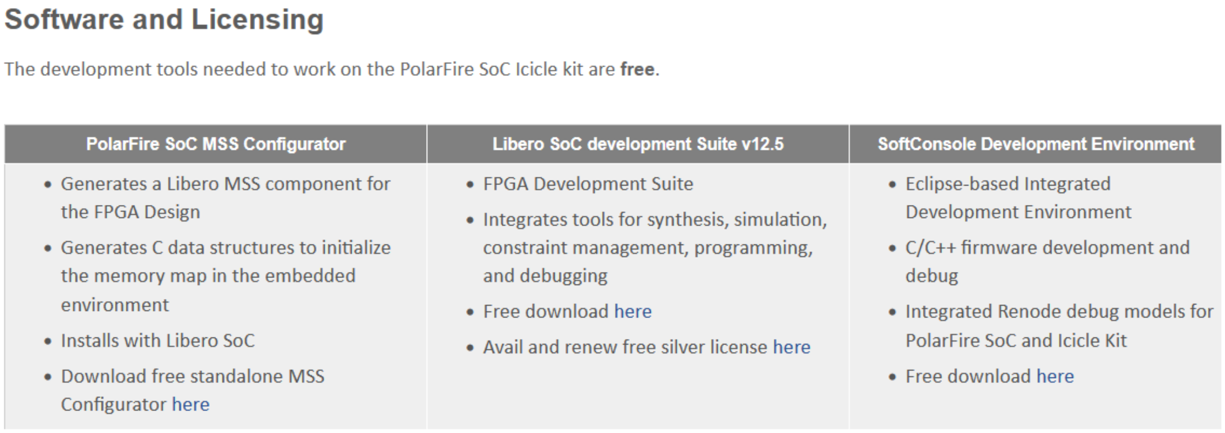
Load, save or delete a stored session

Saved Sessions:

gap8_jtag_usb0
genesys2
genesys2_usb1
maixpy
ttyusb0_115200
ttyusb1_115200

Close window on exit: ● Always ○ Never ○ Only on clean exit

Figure 4: Putty Configuration

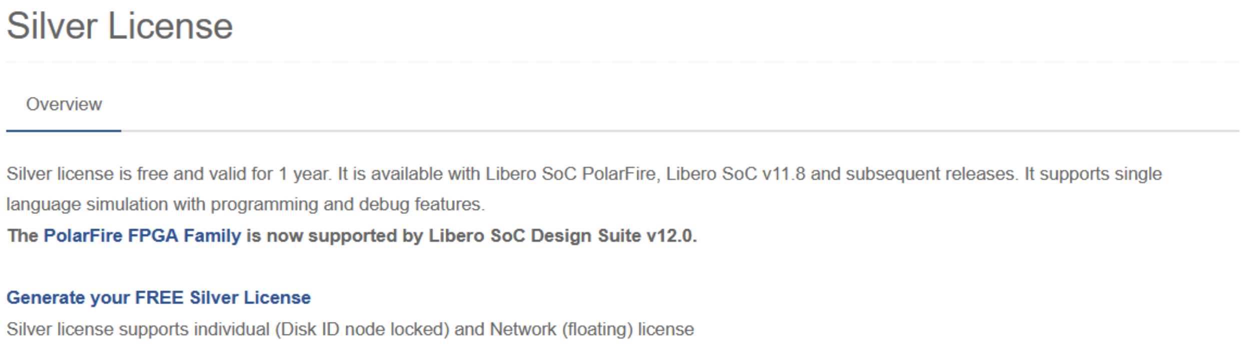


Software and Licensing

The development tools needed to work on the PolarFire SoC Icicle kit are **free**.

PolarFire SoC MSS Configurator	Libero SoC development Suite v12.5	SoftConsole Development Environment
• Generates a Libero MSS component for the FPGA Design • Generates C data structures to initialize the memory map in the embedded environment • Installs with Libero SoC • Download free standalone MSS Configurator here	• FPGA Development Suite • Integrates tools for synthesis, simulation, constraint management, programming, and debugging • Free download here • Avail and renew free silver license here	• Eclipse-based Integrated Development Environment • C/C++ firmware development and debug • Integrated Renode debug models for PolarFire SoC and Icicle Kit • Free download here

Figure 5: Dependencies to get various SW



Silver License

Overview

Silver license is free and valid for 1 year. It is available with Libero SoC PolarFire, Libero SoC v11.8 and subsequent releases. It supports single language simulation with programming and debug features.

The PolarFire FPGA Family is now supported by Libero SoC Design Suite v12.0.

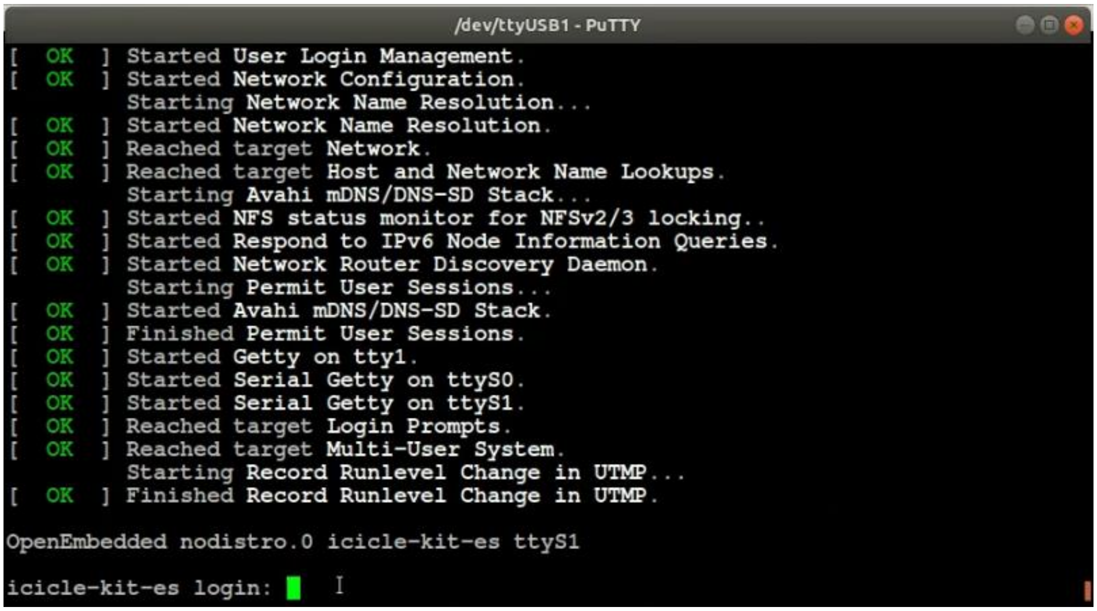
Generate your FREE Silver License

Silver license supports individual (Disk ID node locked) and Network (floating) license

Figure 6: License needed for getting started

Steps to run the code on PolarFire SoC ICICLE ES Kit

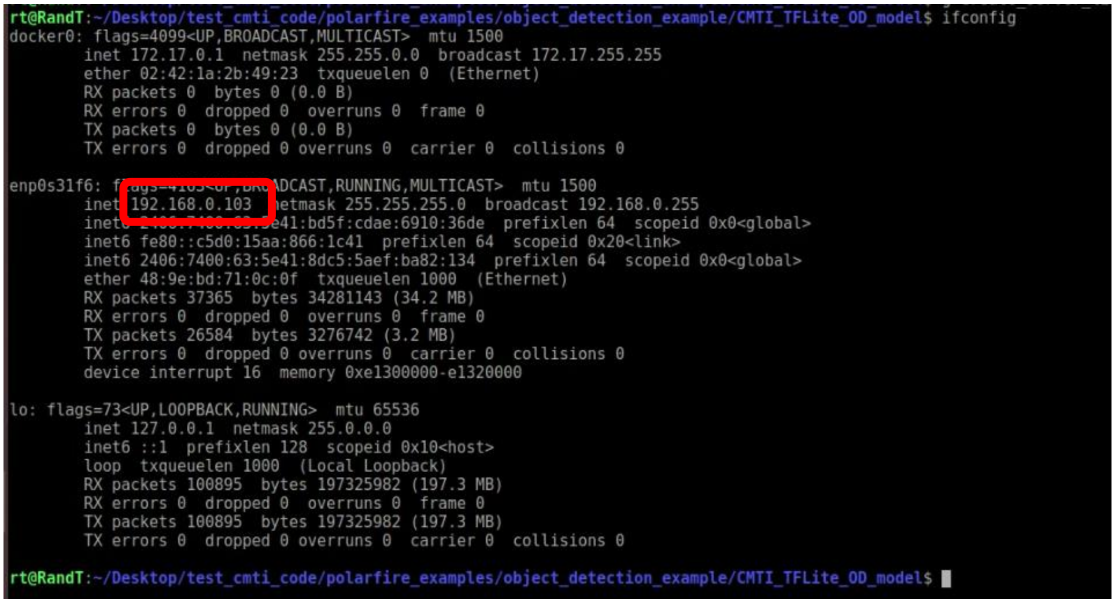
- Connect the power cable.
- Connect the USB-UART cables.
- Connect the open LAN to the Ethernet port on the kit.
- Power on the board. Board will boot the Linux and asks for login password.
  Password – **root**



```
[  OK  ] Started User Login Management.
[  OK  ] Started Network Configuration.
         Starting Network Name Resolution...
[  OK  ] Started Network Name Resolution.
[  OK  ] Reached target Network.
[  OK  ] Reached target Host and Network Name Lookups.
         Starting Avahi mDNS/DNS-SD Stack...
[  OK  ] Started NFS status monitor for NFSv2/3 locking..
[  OK  ] Started Respond to IPv6 Node Information Queries.
[  OK  ] Started Network Router Discovery Daemon.
         Starting Permit User Sessions...
[  OK  ] Started Avahi mDNS/DNS-SD Stack.
[  OK  ] Finished Permit User Sessions.
[  OK  ] Started Getty on tty1.
[  OK  ] Started Serial Getty on ttyS0.
[  OK  ] Started Serial Getty on ttyS1.
[  OK  ] Reached target Login Prompts.
[  OK  ] Reached target Multi-User System.
         Starting Record Runlevel Change in UTMP...
[  OK  ] Finished Record Runlevel Change in UTMP.

OpenEmbedded nodistro.0 icicle-kit-es ttyS1

icicle-kit-es login:
```

Figure 7: Kit login password

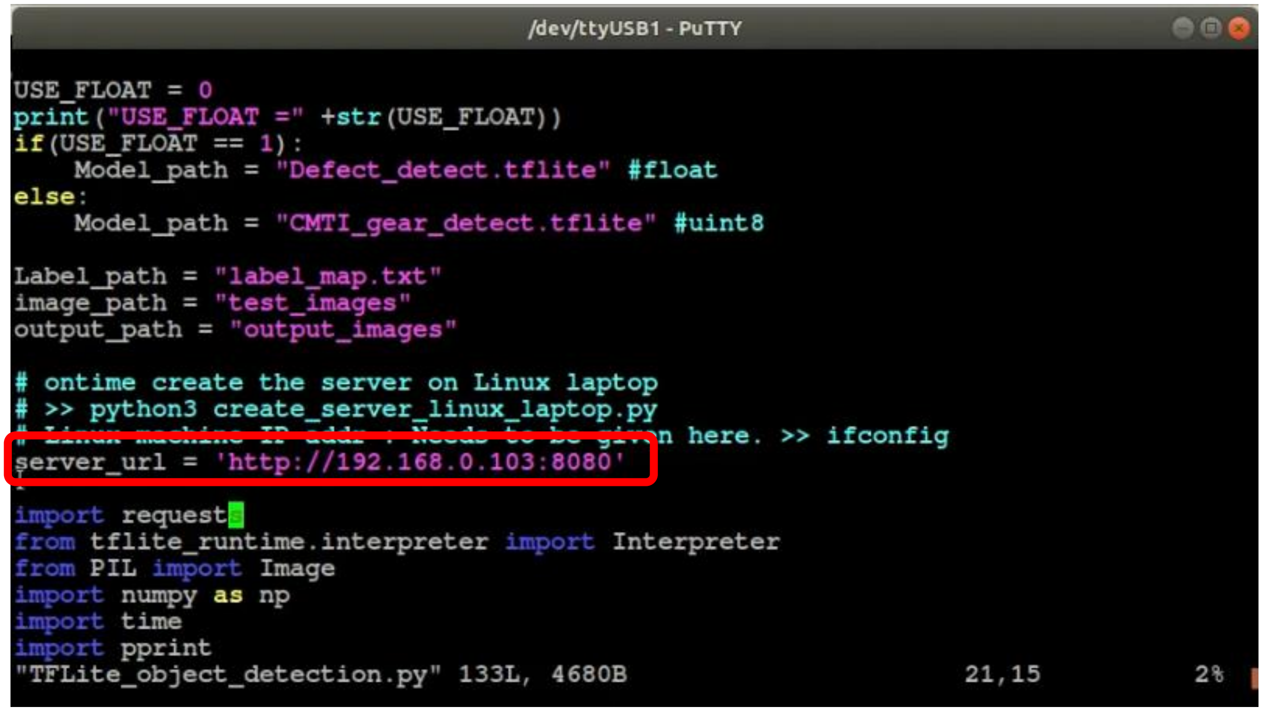
- On laptop:
    - $git clone < https://github.com/anjaliigedam/polarfire_examples.git>
    - $cd ip_camera_object_detection
    - Check IP address:
      $ifconfig



```
rt@RandT:~/Desktop/test_cmti_code/polarfire_examples/object_detection_example/CMTI_TFLite_OD_model$ ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:1a:2b:49:23  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s31f6: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.103  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 2406:7400:63:5e41:bd5f:cdae:6910:36de  prefixlen 64  scopeid 0x0<global>
        inet6 fe80::c5d0:15aa:866:1c41  prefixlen 64  scopeid 0x20<link>
        inet6 2406:7400:63:5e41:8dc5:5aef:ba82:134  prefixlen 64  scopeid 0x0<global>
        ether 48:9e:bd:71:0c:0f  txqueuelen 1000  (Ethernet)
        RX packets 37365  bytes 34281143 (34.2 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 26584  bytes 3276742 (3.2 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 16  memory 0xe1300000-e1320000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 100895  bytes 197325982 (197.3 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 100895  bytes 197325982 (197.3 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

rt@RandT:~/Desktop/test_cmti_code/polarfire_examples/object_detection_example/CMTI_TFLite_OD_model$
```

Figure 8: IP address of Laptop

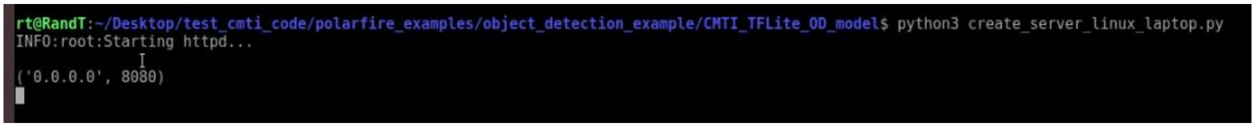
- Set this laptop IP address in the script (TFLite_object_detection.py).



```
USE_FLOAT = 0
print("USE_FLOAT =" +str(USE_FLOAT))
if(USE_FLOAT == 1):
    Model_path = "Defect_detect.tflite" #float
else:
    Model_path = "CMTI_gear_detect.tflite" #uint8

Label_path = "label_map.txt"
image_path = "test_images"
output_path = "output_images"

# ontime create the server on Linux laptop
# >> python3 create_server_linux_laptop.py
# Linux machine IP addr : Needs to be given here. >> ifconfig
server_url = 'http://192.168.0.103:8080'

import requests
from tflite_runtime.interpreter import Interpreter
from PIL import Image
import numpy as np
import time
import pprint
"TFLite_object_detection.py" 133L, 4680B                    21,15         2%
```

Figure 9: IP address setting

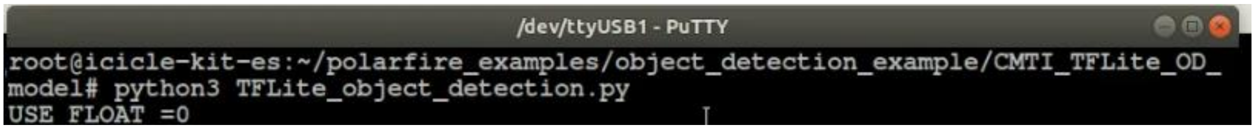
- Run server script:
  $python3 create_server_linux_laptop.py



```
rt@RandT:~/Desktop/test_cmti_code/polarfire_examples/object_detection_example/CMTI_TFLite_OD_model$ python3 create_server_linux_laptop.py
INFO:root:Starting httpd...

('0.0.0.0', 8080)
```

Figure 10: Run server script in Laptop

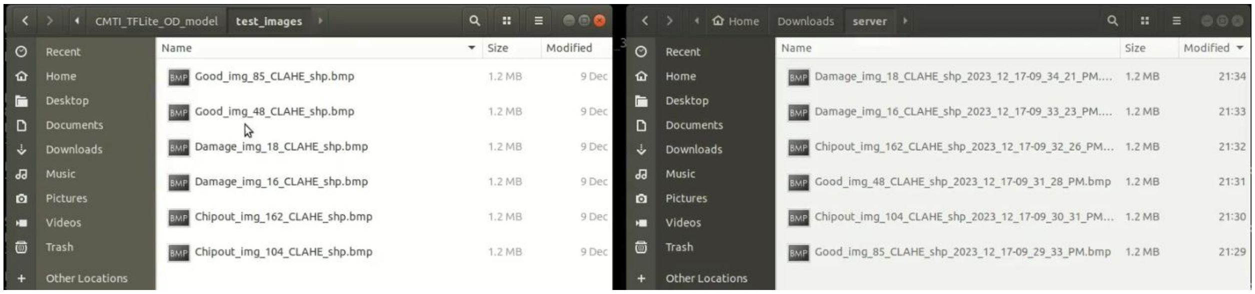
- On kit:
  - $git clone < https://github.com/anjaliigedam/polarfire_examples.git>
  - $cd ip_camera_object_detection
  - Check the IP address:
    $ifconfig
    Set kit and laptop IP address in the script (TFLite_object_detection.py).
  - Run defect detection script:
    $python3 TFLite_object_detection.py



```
root@icicle-kit-es:~/polarfire_examples/object_detection_example/CMTI_TFLite_OD_
model# python3 TFLite_object_detection.py
USE_FLOAT =0
```

Figure 11: Run defect detection script on Kit

- TFLite_object_detection.py script takes input from **test_images** and dumps the corresponding output to **Downloads/server/** directory created by this script with timestamp.



Name	Size	Modified
Good_img_85_CLAHE_shp.bmp	1.2 MB	9 Dec
Good_img_48_CLAHE_shp.bmp	1.2 MB	9 Dec
Damage_img_18_CLAHE_shp.bmp	1.2 MB	9 Dec
Damage_img_16_CLAHE_shp.bmp	1.2 MB	9 Dec
Chipout_img_162_CLAHE_shp.bmp	1.2 MB	9 Dec
Chipout_img_104_CLAHE_shp.bmp	1.2 MB	9 Dec

Name	Size	Modified
Damage_img_18_CLAHE_shp_2023_12_17-09_34_21_PM....	1.2 MB	21:34
Damage_img_16_CLAHE_shp_2023_12_17-09_33_23_PM....	1.2 MB	21:33
Chipout_img_162_CLAHE_shp_2023_12_17-09_32_26_PM...	1.2 MB	21:32
Good_img_48_CLAHE_shp_2023_12_17-09_31_28_PM.bmp	1.2 MB	21:31
Chipout_img_104_CLAHE_shp_2023_12_17-09_30_31_PM...	1.2 MB	21:30
Good_img_85_CLAHE_shp_2023_12_17-09_29_33_PM.bmp	1.2 MB	21:29

Figure 12: Input and output images

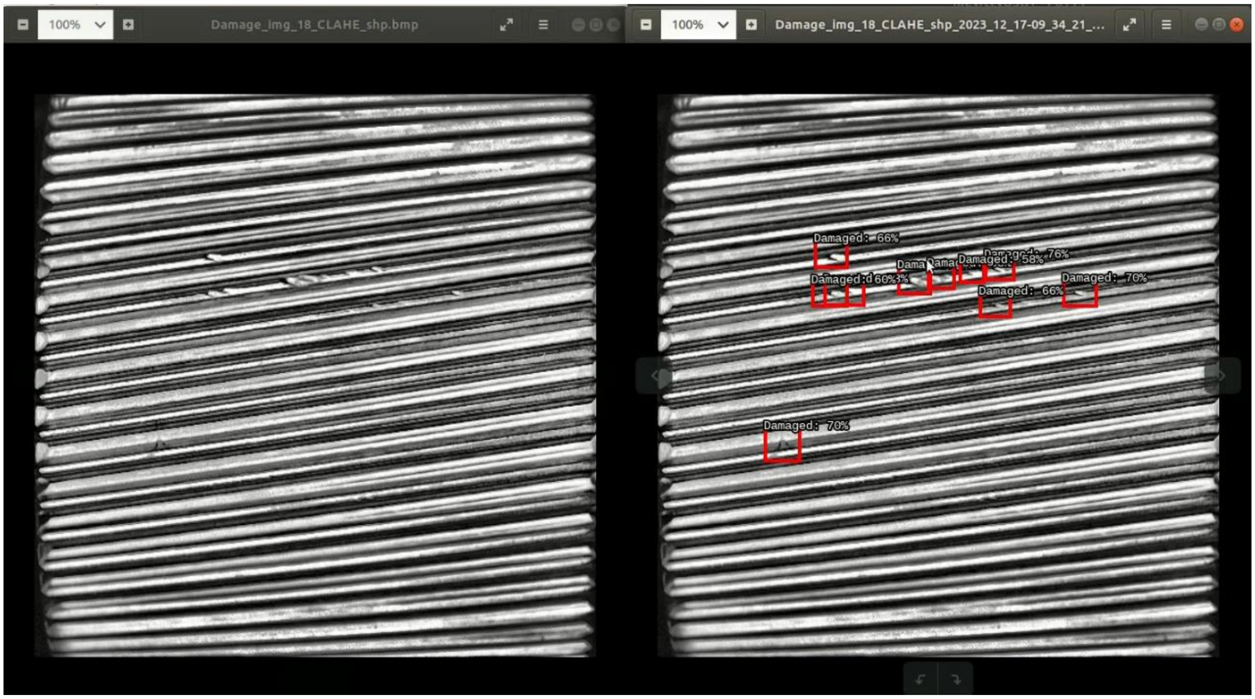


Figure 13: Input and output images