## Programming Talaria TWO

### Hardware Set-up

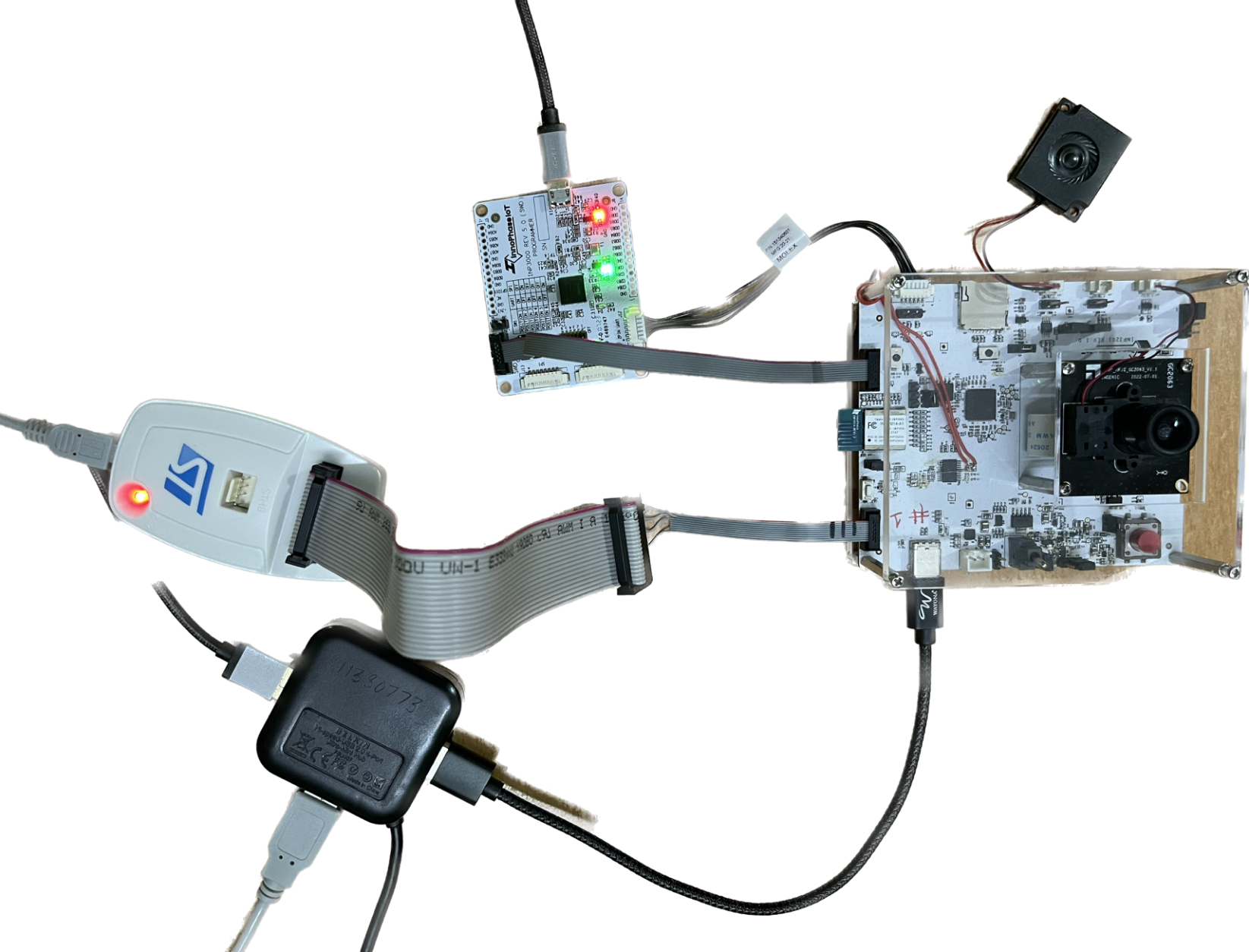


Figure 8: Hardware set-up

**Note**:

1. For more details on the hardware requirements, refer section: *Hardware Requirements* of the document: UG\_Dual-Stack.pdf (*talaria\_two\_dual\_stack\_vx.y\host\INP3201\doc*).
2. For pin-outs details on INP3000 programmer board, refer UG\_Programming\_using\_INP3000.pdf (*freertos\_sdk\_x.y\doc\user\_guides\ug\_programming\_using\_INP3000*).

### Procedure

Program the Dual-Stack ELF (<dual\_stack\_sdio>.elf) onto Talaria TWO using the following steps:

1. Open a terminal on console machine and start openocd from FreeRTOS SDK root folder *(freertos\_sdk\_x.y\)*. This enables in-system programming on Talaria TWO.

Command:

|  |
| --- |
| sudo openocd -s ./conf/ -f ftdi\_swd.cfg -f t2\_swd.cfg |

Expected output:

|  |
| --- |
| Open On-Chip Debugger 0.10.0  Licensed under GNU GPL v2  For bug reports, read  http://openocd.org/doc/doxygen/bugs.html  adapter speed: 1000 kHz  Info : FTDI SWD mode enabled  cortex\_m reset\_config sysresetreq  Info : clock speed 1000 kHz  Info : SWD DPIDR 0x2ba01477  Info : talaria\_two.cpu: hardware has 6 breakpoints, 4 watchpoints  Info : Talaria TWO rev. H0  cortex\_m reset\_config sysresetreq Info : clock speed 1000 kHz Info : SWD DPIDR 0x2ba01477 Info : talaria\_two.cpu: hardware has 6 breakpoints, 4 watchpoints Info : Talaria TWO rev. H0 |

1. Start arden.py script from SDK root folder *(sdk\_x.y\)* on a new terminal:

Command:

|  |
| --- |
| sudo python3 ./script/arden.py ./apps/gordon-jtag/bin/gordon-jtag.elf |

Expected output:

|  |
| --- |
| [sudo] password for dell10:  (re)connecting with openocd  openocd version: 'Open On-Chip Debugger 0.10.0'  boot rom version of connected device: "yoda-h0-rom-16-0-gd5a8e586"  starting up on localhost port 10000 |

1. Run the script gdbrun.py from FreeRTOS SDK root folder *(freertos\_sdk\_x.y\)* on a new terminal. Given a HOST with running OpenOCD and UART relay, gdbrun.py connects to Host and load/boot an ELF image.

Command:

|  |
| --- |
| sudo python3 ./script/gdbrun.py ./apps/gordon-jtag/bin/gordon-jtag.elf --noconsole --nowait |

1. The <dual\_stack\_sdio>.elf generates dualstack.img and dualstack.img.vm and they are flashed to Talaria TWO using the following sequence of commands.

Command:

|  |
| --- |
| sudo python3 ./script/boot.py --output dualstack.img ./solutions/dual\_stack/bin/dual\_stack\_sdio.elf hio.transport=sdio hio.maxsize=8192 hio.sdio\_mhz=10 wifi.outq\_max=32 hio.irq\_min\_gap=60 hio.irq\_retry\_time=200 wifi.rts=2 wifi.pmode\_cts=1 |

**Note**:

* 1. To enable Host reboot when Talaria TWO reboots, use the following bootargs:

|  |
| --- |
| krn.gpio=--------------p----p-p ds.host\_pwroff\_gpio\_state=LLLLLL--------p--p--p- |

* 1. For video streaming, use the following bootargs:

|  |
| --- |
| hio.min\_heap\_for\_burst\_tx=24000 hio.max\_data\_ind=3 |

1. The minimum heap required for burst mechanism is 24000.
2. hio.max\_data\_ind specifies the maximum number of packets which can be queued in HIO for transmission.

Expected output:

|  |
| --- |
| Loading ./solutions/dual\_stack/bin/dual\_stack\_sdio.elf  Entrypoint 0xb1159  Wrote bootimage to  .img 103\_972 bytes  Wrote vmimage to dualstack.img.vm 482\_176 bytes |

Command:

|  |
| --- |
| ./script/flash.py --device localhost:10000 part\_write BOOT @root/freertos\_sdk\_x.y/dualstack.img |

**Note**: Replace x and y in freertos\_sdk\_x.y with the appropriate SDK release version.

Expected output:

|  |
| --- |
| write: [==================================================================>] 103972 bytes written to addr 4096 |

Command:

|  |
| --- |
| ./script/flash.py --device localhost:10000 part\_write VIRT @root/freertos\_sdk\_x.y/dualstack.img.vm |

Expected output:

|  |
| --- |
| write: [==================================================================>] 482176 bytes written to addr 262144 |

After flashing the dual\_stack\_sdio.elf, update the root.img with AWS certificates on /data partition.

Copy the AWS IoT certificates (aws\_device\_cert.crt, aws\_device\_pkey.key & aws\_root\_ca.crt) into the rootfs folder (*freertos\_sdk\_x.y/root\_fs/root*) in the SDK package and use mklittlefs command to create a new root image.

|  |
| --- |
| ./tools/mklittlefs/mklittlefs -s 0x40000 -c ./root\_fs/root root.img |

Command:

|  |
| --- |
| ./pc\_tools/T2\_Flasher/bin/T2\_Flasher\_Linux --operation=write\_part --partition=DATA ./root.img |

Expected output:

|  |
| --- |
| /tmp/\_MEItbY8QH/bin  Starting OpenOCD  /tmp/\_MEItbY8QH/bin/openocd/linux64/openocd\_linux64 -s /tmp/\_MEItbY8QH/bin/sdk-files -f /tmp/\_MEItbY8QH/bin/openocd/interface/cmsis-dap.cfg -f t2\_swd.cfg  xPack OpenOCD, x86\_64 Open On-Chip Debugger 0.10.0+dev (2020-10-13-17:27)  Licensed under GNU GPL v2  For bug reports, read  http://openocd.org/doc/doxygen/bugs.html  DEPRECATED! use 'adapter speed' not 'adapter\_khz'  cortex\_m reset\_config sysresetreq  Error: couldn't bind tcl to socket on port 6666: Address already in use  (re)connecting with openocd  Info : accepting 'tcl' connection on tcp/6666  talaria\_two  Open On-Chip Debugger 0.10.0  openocd version: 'Open On-Chip Debugger 0.10.0'  Connecting GDB  Info : accepting 'gdb' connection on tcp/3333  GDB: monitor reset halt  Info : Talaria TWO rev. H0  target halted due to debug-request, current mode: Thread  xPSR: 0x01000000 pc: 0x00020f90 msp: 0x00041a78  GDB: gdb load /tmp/\_MEItbY8QH/bin/tinyflasher.elf  GDB: write WAIT\_OP at 0x7ffe8  Starting TinyFlasher  GDB: gdb continue  GDB: write READ\_PTABLE\_OP at 0x7ffe8  GDB: gdb continue  waiting...  operation complete  GDB: write WAIT\_OP at 0x7ffe8  Read ptable contents from memory  Flashing ./root.img at 0x100000  GDB: write sector count at 0x7fff0  GDB: write target flash addr at 0x7fff4  GDB: write image binary to RAM staging location (0x80000)  GDB: write FLASH\_OP at 0x7ffe8  GDB: gdb continue  waiting...  operation complete  GDB: write WAIT\_OP at 0x7ffe8  Disconnecting OpenOCD  DONE  Info : dropped 'tcl' connection  Info : dropped 'gdb' connection |

Once flashing and root image update is complete, reset Talaria TWO and reboot T31ZX to start the demo application.