

# Intel Atom<sup>®</sup> Processor C3200RK Series for Yocto Project\*

Release Notes (MR4.1)

---

*October 2017*

*Revision 011US*



You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting:  
<http://www.intel.com/design/literature.htm>.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at <http://www.intel.com/> or from the OEM or retailer.

No computer system can be absolutely secure.

Intel, Intel Atom, Intel Core, and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

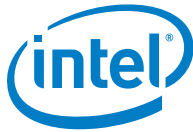
\*Other names and brands may be claimed as the property of others.

Copyright © 2017, Intel Corporation. All rights reserved.



# Contents

<b>1.0</b>	<b>Introduction.....</b>	<b>6</b>
1.1	Intended Audience.....	6
1.2	Customer Support.....	6
1.3	Terminology.....	6
1.4	Reference Documents.....	7
<b>2.0</b>	<b>Release Summary.....</b>	<b>9</b>
2.1	Release Content.....	9
2.2	Supported Operating Systems.....	9
2.3	Hardware and Software Compatibility.....	9
2.4	Major Features.....	11
2.5	New Features.....	21
2.6	Limitations.....	21
2.7	Known Issues.....	21
2.8	Fixed Issues.....	21
2.9	Where to Find the Release .....	22
<b>3.0</b>	<b>Setting up Build Environments .....</b>	<b>23</b>
3.1	Setting up the Host Machine.....	23
3.1.1	Proxy Server Configuration for Internet Access.....	23
3.1.2	Installing Additional Packages.....	24
3.1.3	Setting Up git Configuration for Open Source GIT Repository.....	25
3.2	Installing the Build System.....	26
<b>4.0</b>	<b>Compiling the Build .....</b>	<b>27</b>
4.1	Running the Setup Script.....	27
4.2	Configure the BSP to Enable Different Platform and Display Support .....	27
4.3	Enabling Wayland* Protocol as Default Windowing System .....	31
4.3.1	Launching Wayland Protocol on Target System .....	33
4.3.2	Integrating and Testing Weston* Applications.....	33
4.4	Enabling X Window System* (X11) as Default Windowing System .....	34
4.4.1	Testing X Window System (X11) on SoFIA 3G R.....	35
4.5	Optimizing <code>conf/local.conf</code> .....	35
<b>5.0</b>	<b>Flashing Binaries on Target Platform .....</b>	<b>37</b>
5.1	Flashing Instructions with the Intel® Platform Flash Tool (Intel® PFT) Using Windows* Host .....	38
5.2	Flash Binary Packaging Script.....	45
<b>6.0</b>	<b>Video Playback Using GStreamer* Framework.....</b>	<b>46</b>
6.1	Enabling Audio on SoFIA .....	46
	<b>Appendix A.....</b>	<b>48</b>



A.1	Workaround for Issue 1804305651 .....	48
A.2	Workaround for Issue 1504330812 .....	50

## Figures

Figure 1.	TOPDIR Usage in conf/local.conf .....	36
Figure 2.	Flashing Android* OS with "SMP FLS only" Flashing Option.....	38
Figure 3.	Intel® Platform Flash Tool (Intel® PFT) Desktop Icon.....	39
Figure 4.	Intel® Platform Flash Tool (Intel® PFT) Window .....	40
Figure 5.	Browse to the fls Zip File.....	41
Figure 6.	Flashing with "SMP FLS Only".....	42
Figure 7.	Download Progress Screen.....	43
Figure 8.	Download Success Screen.....	44

## Tables

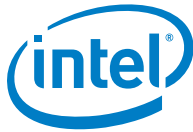
Table 1.	Terminology.....	6
Table 2.	Reference Documents .....	7
Table 3.	Release Content .....	9
Table 4.	Release Information .....	9
Table 5.	Hardware Configuration (Telit* HE922-3GR Hardware) .....	10
Table 6.	Hardware Configuration (Daggett Island/ Halifax River) .....	10
Table 7.	Supported Hardware and Display Interface Combination .....	11
Table 8.	Major Features .....	11
Table 9.	Known Issues.....	21
Table 10.	Fixed Issues – Others .....	21
Table 11.	Fixed Issues - Communications.....	22



## Revision History

---

Date	Revision	Description
October 2017	011	<ul style="list-style-type: none"><li>MR4.1 release</li><li>Updated available audio path (page18)</li></ul>
July 2017	010	MR4.0 release
June 2017	009	MR3.4 release
June 2017	008	MR3.2 release
May 2017	007	MR3.1 release
April 2017	006	MR3.0 release
March 2017	005	MR2.1 release
February 2017	004	Updated Section 5
January 2017	003	MR2 release
---	002	Skipped version
November 2016	001	Initial release (MR1)



## 1.0 Introduction

---

The Board Support Packages (BSPs), also known as the BSPs, provide selected tune options and generic hardware support for most current Intel® processors and Intel® products. In this release, Intel provides the Yocto Project\*-based Board Support Package for Intel Atom® C3200RK processor series for testing and project-based software development purposes.

### 1.1 Intended Audience

This Release Note is intended for customers who want to use Intel Atom® C3200RK Processor Series for Yocto Project\*.

### 1.2 Customer Support

For hardware-related support, contact Telit\* for more details.

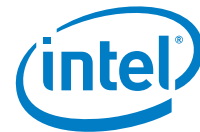
For Software-related support, contact software partners enabled by Intel for additional support information; or your Intel representation for more details.

Refer to <https://www-ssl.intel.com/content/www/us/en/embedded/products/sofia-3g-r/overview.html> for product details.

### 1.3 Terminology

**Table 1. Terminology**

Term	Description
AFE	Analog Front End
BP	Baseband Processor
BSP	Board Support Package (this software)
DDK	Driver Development Kit
DSP	Digital Signal Processing
DMA	Direct Memory Access
eMMC*	Embedded Multi-Media Card
FOSS	Free Open Source Software
GNSS	Global Navigation Satellite System



Term	Description
IBP	Intel Business Portal, <a href="https://businessportal.intel.com/irj/portal">https://businessportal.intel.com/irj/portal</a>
IIO	Industrial I/O
IoT	Internet of Things
LVDS	Low-voltage differential signaling
MMC	Multimedia Card
MR	Maintenance Release
MIPI*-DSI	Mobile Industry Processor Interface - Display Serial Interface
NVM	Non-Volatile Memory
OMA	Open Mobile Alliance
OS	Operating System
PID	Process ID
PIO	Programmable Input/Output
PRH	Procedure Request Handler
RNDIS	Remote Network Driver Interface Specification
RPC	Remote Procedure Call
SRC	Sample Rate Conversion
UART	Universal Asynchronous Receiver/Transmitter
USIF	Universal Serial Interface

## 1.4 Reference Documents

**Table 2. Reference Documents**

Document	Document No./Location
<i>Intel Atom® X3-C3200RK SoFIA Yocto Project* - Getting Started Guide</i>	564478
<i>Intel Atom® X3-C3200RK SoFIA Yocto Project* Communication Support – Getting Started Guide</i>	567013
<i>PlatformFlashTool_5g8.5.1.0_win32.zip (For Windows* OS)</i>	566922
<i>Platformflashtool_5.5.1.0_linux_x86_64.zip (For Linux* OS)</i>	566919
<i>PlatformFlashTool_5.5.1.0_mac64.zip (For Mac* 64 OS)</i>	566921
<i>Android* OS with Board Support Package for Intel Atom® x3-C3200RK Processor (Formerly SoFIA) Release Notes</i>	560062
<i>Intel® Phone Flash Tool Installer</i>	560547



Document	Document No./Location
<i>Intel Atom® x3-C3200RK Processor (formerly SoFIA) MoW</i>	569878

§





## Release Summary

This revision is BSP MR4.1 version for Intel Atom® C3200RK Processor series that support different combinations of hardware, with and without display interfaces.

### 1.5 Release Content

Table 3. Release Content

Type	File Name / Description
Release Document	Intel Atom® Processor C3200RK Series for Yocto Project* Release Notes 335251-011US_IntelAtom_C3200_Yocto_Project_Release_Notes_MR4_1.pdf
Release Packages	sofia3gr_mr4.1_linux-only-20171012_132501.tar.bz2 sofia3gr_mr4.1_linux-only-20171012_132501.tar.bz2.md5sum

### 1.6 Supported Operating Systems

This release only supports Yocto Project\*.

### 1.7 Hardware and Software Compatibility

Table 4. Release Information

Type	Description
Release Version	MR4.1
Software Version	MR4.1Y_170905, 08
Yocto Project* Version	Yocto Project* v2.0.2 (Jethro*)-based project: openembedded-core, BitBake, meta-openembedded, and meta-intel-iot-security
Kernel Version	Linux* v3.14.55 based kernel patch-series
Modem Version	SF_3GR_MAINT_01.1709.05_SP_MB



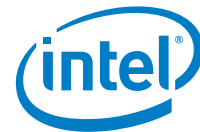
**Table 5. Hardware Configuration (Telit\* HE922-3GR Hardware)**

Type	Description
Board	Telit* HE922/WE922-3GR with J95Q
CPU	CPU VR NCP6336ZB, Telit HE922 built-in
eMMC*	Telit HE922/WE922-3GR Built-in
Display Panel	Telit Add-On-Board (MIPI-DSI)
Touch Screen Display Panel	Telit HE922/WE922-3GR Built-in
Speaker and Digital Mic	Telit HE922/WE922-3GR River Built-in
Sensor	N/A
Memory	Telit HE922/WE922-3GR Built-in
SD* Card	Kingston* microSDHC (SDC4/4 GB)

**Note:** For certification info, refer to the *Intel Atom® x3-C3200RK Processor (formerly SoFIA) MoW*.

**Table 6. Hardware Configuration (Daggett Island/ Halifax River)**

Type	Description
Board	Daggett Island/Halifax River Fab A and Fab B
CPU	Intel Atom® processor C3200 series
eMMC	Daggett Island Built-in
Display Panel	Halifax River Add-On-Board (MIPI-DSI)/ LVDS Add-On-Board Sharp LQ070Y3LG05
Touch Screen Display Panel	Halifax River Built-in
Speaker and Digital Microphone	Halifax River Built-in
Sensor	Halifax River Built-in (Accelerometer, Compass, Gyroscope, Ambient Light, and Proximity)
Memory	Daggett Island Built-in 1 GB
SD Card	Kingston* microSD HC (SDC4/4 GB)



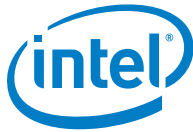
**Table 7. Supported Hardware and Display Interface Combination**

Hardware	Display Interface
Wilbur Bay, Telit HE922, Telit WE922	MIPI-DSI
Wilbur Bay	LVDS Add-On-Board Sharp LQ070Y3LG05
Wilbur Bay, Telit HE922, Telit WE922	Headless (no display)

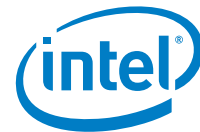
## 1.8 Major Features

**Table 8. Major Features**

Number	Feature
1	<b>AT+CGMR Command</b> This AT command will return Software version in “MRx.xY_17WWSS, SVN” format. <ul style="list-style-type: none"><li>• First four digits indicate MR version e.g., MR3.0</li><li>• The letter Y indicates OS Yocto version</li><li>• WWSS indicates modem version week number and sub-week/day number</li><li>• SVN indicates software version number</li></ul>
2	<b>Audio Playback and Record</b> <ul style="list-style-type: none"><li>• alsa-lib v1.0.29</li><li>• alsa-utils v1.0.29</li><li>• Supports 48 kHz sample rate at the DSP frontend for playback and recording</li><li>• Supports 48 kHz sample rate at the DSP backend for playback and recording</li><li>• Supports 16-bit Bit-depth</li><li>• Supports stereo channel for playback and record</li></ul> <b>Supports Agold620 AFE Loudspeaker, AFE Earphone, AFE Digital Microphone, and AFE SRC 48 kHz</b>
3	<b>GPIO and Pin-Mux</b> <ul style="list-style-type: none"><li>• Pad Control Logic digital pads GPIO<ul style="list-style-type: none"><li>— pinctrl driver</li><li>— GPIO pin layout defined through device-tree</li></ul></li></ul>
4	<b>I<sup>2</sup>C*</b> <ul style="list-style-type: none"><li>• I<sup>2</sup>C* v2.1 January 2000 specification</li><li>• Data transfer mode: Fast (400-kBaud) and standard (100-kBaud)</li></ul>



Number	Feature
	<ul style="list-style-type: none"> <li>Master, multi-master, and slave role</li> </ul>
5	<b>MMC SD Host Controller</b> <ul style="list-style-type: none"> <li>Supports embedded multi-media card (eMMC) <ul style="list-style-type: none"> <li>eMMC Specification v4.51</li> </ul> </li> <li>1-, 4-, and 8-bit (1.8 V) mode <ul style="list-style-type: none"> <li>Up to 832Mbps data rate (8-bit, eMMC Specification v4.51 DDR (Class-J) @ 52 MHz)</li> <li>Supports SD Card and SDIO* interface</li> <li>SD Host Controller Standard Specification v3.0</li> <li>SDIO Card Specification v3.0</li> <li>SD Memory Card Specification Draft v 3.0</li> <li>SD Memory Card Security Specification v1.01</li> <li>1- and 4-bit SD mode</li> <li>Up to 384 Mbps data rate (4-bit: SD3.0 DDR50 mode @ 48 MHz, SDR50 @ 96 MHz)</li> </ul> </li> </ul>
6	<b>USB v2.0</b> <ul style="list-style-type: none"> <li>On-The-Go Dual Role Device controller (device or host mode)</li> <li>Host mode: High Speed (480 MB/s), Full Speed (12 MB/s), and Low Speed (1.5 MB/s)</li> <li>Device mode: High Speed (480 MB/s) and Full Speed (12 MB/s)</li> <li>USB keyboard and mouse</li> <li>Remote Network Driver Interface Specification (RNDIS) driver for Ethernet connection over USB (client mode)</li> <li>Battery Charging Specification Rev 1.2</li> </ul>
7	<b>USIF-SPI</b> <ul style="list-style-type: none"> <li>Both PIO and DMA transfer mode</li> <li>Character length = 8-, 16- and 32-bit</li> <li>Shift direction: LSB or MSB shifted out first</li> <li>Master/Slave mode</li> </ul>
8	<b>USIF-UART</b> <ul style="list-style-type: none"> <li>Both PIO and DMA transfer mode</li> <li>16C550 standard</li> <li>Default: 8-bit, no flow-control, no parity check and 115200</li> <li>UART controller kernel clock-rate range: 48-, 96-, 104-, and 260-kHz</li> </ul>
9	<b>Thermal daemon with XML-based Thermal Policy Management</b>



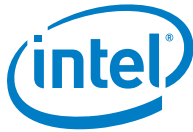
Number	Feature
	<ul style="list-style-type: none"> <li>• CPU zone monitoring with CPU frequency scaling</li> <li>• Battery zone monitoring with charging current throttling</li> <li>• Platform shut-down during critical event</li> <li>• Throttling methods:               <ul style="list-style-type: none"> <li>— Maximum trip type with PID control (compensation increases step wise and then exponentially)</li> <li>— Absolute control value settings whereby each trip point is bound to specific cooling states of the cooling device</li> </ul> </li> </ul>
10	<p><b>meta-intel-iot-middleware ready ingredients (for IoT OS)</b></p> <ul style="list-style-type: none"> <li>• Libmraa with IIO trigger buffer and IIO event extension for Daggett Island/Halifax River Fab A and Fab X platforms</li> <li>• UPM driver and application examples for the following I<sup>2</sup>C*-based sensors:               <ul style="list-style-type: none"> <li>— MEMSIC* MMC35240 magnetic sensor</li> <li>— STMicroelectronics* L3GD20 gyroscope sensor</li> </ul> </li> <li>• Kionix* KXCJK-1013 accelerometer</li> <li>• Avago Technologies* APDS-9930 light and proximity sensor</li> <li>• Sensor uses mount-matrix compliant to IIO Core</li> </ul>
11	<p><b>Comms: Cellular</b></p> <ul style="list-style-type: none"> <li>• This is a release for Comms Cellular enabling through software implementation for achieving <b>PS Data Call</b>.</li> <li>• Radio Interface Layer Daemon (RILD)               <ul style="list-style-type: none"> <li>— Radio Interface Layer Daemon initializes the Vendor RIL, processes all communication from Framework telephony services, and dispatches calls to the Vendor RIL as solicited commands.</li> </ul> </li> <li>• Vendor RIL (VRIL)               <ul style="list-style-type: none"> <li>— The radio-specific Vendor RIL of ril.h that processes all communication with radio hardware and dispatches calls to the RIL daemon (rild) through unsolicited commands. Vendor Radio Interface Layer provides an abstraction layer between telephony services and radio hardware.</li> </ul> </li> <li>• RPC               <ul style="list-style-type: none"> <li>— The RPC implements modem-specific procedures, called the "aggregator function library" to handle RIL requests, responses, and indications. The RPC that runs on the AP can call the UTA APIs on the BP.</li> </ul> </li> </ul>



Number	Feature
	<ul style="list-style-type: none"><li>• OFONO<ul style="list-style-type: none"><li>— oFono provides a mobile telephony application development framework that includes APIs to interact with RILD.</li></ul></li><li>• OSAL<ul style="list-style-type: none"><li>— Creates operating system abstraction to remove Android* OS dependency and replacing it with a Linux* OS equivalent.</li></ul></li><li>• LIBNVMMODULE<ul style="list-style-type: none"><li>— Maintains NVM data used by the platform.</li><li>— Supports WCDMA Band 1, Band 2, Band 5, and Band 8.</li><li>• Support for OMA Device Management IMEI Sync</li></ul></li></ul>



Number	Feature
12	<p data-bbox="561 321 805 352"><b>Comms: Bluetooth®</b></p> <ul data-bbox="578 363 1401 1717" style="list-style-type: none"><li data-bbox="578 363 1401 436">• This is a release for Comms Bluetooth® enabling through software implementation for Bluetooth® communication.</li><li data-bbox="578 447 1401 520">• Linux* Kernel Backport version (Bluetooth® driver and subsystem only): version 4.3.</li><li data-bbox="578 531 1401 699">• Remote Host Wakeup feature is now supported. Bluetooth® Host that was suspended due to inactivity can now be woken up by some activity triggered on the remote connected device (for example, connected Bluetooth® HID keyboard sending Bluetooth® data, on a key pressed by the user).</li><li data-bbox="578 709 1401 1717">• Packages in BlueZ 5.37<ul style="list-style-type: none"><li data-bbox="610 741 1401 888">— Hciattach<ul style="list-style-type: none"><li data-bbox="659 783 1401 888">○ Attach a serial Universal asynchronous receiver/transmitter (UART) to the Bluetooth® stack as HCI transport interface.</li></ul></li><li data-bbox="610 898 1401 1087">— Hciconfig<ul style="list-style-type: none"><li data-bbox="659 940 1401 972">○ Used to configure Bluetooth® devices</li><li data-bbox="659 982 1401 1014">○ Open/close and initialize HCI devices</li><li data-bbox="659 1024 1401 1087">○ Enable/disable features like page scan/enquiry scan, authentication</li></ul></li><li data-bbox="610 1098 1401 1213">— Bluetoothctl<ul style="list-style-type: none"><li data-bbox="659 1140 1401 1171">○ Scans for remote Bluetooth® devices</li><li data-bbox="659 1182 1401 1213">○ Pairs to remote Bluetooth® devices</li></ul></li><li data-bbox="610 1224 1401 1444">— Sdptool<ul style="list-style-type: none"><li data-bbox="659 1266 1401 1339">○ Provides the interface to perform SDP queries on Bluetooth® devices</li><li data-bbox="659 1350 1401 1444">○ Provides a list of the features of the Bluetooth® device to be handled (local and remote), referenced by the given MAC address</li></ul></li></ul></li><li data-bbox="578 1455 1401 1717">• Other Packages<ul style="list-style-type: none"><li data-bbox="610 1497 1401 1717">— Bluetooth® Driver<ul style="list-style-type: none"><li data-bbox="659 1539 1401 1612">○ New Bluetooth® HCI UART driver implemented for AG6XX devices</li><li data-bbox="659 1623 1401 1717">○ Provides functionality to initialize the Bluetooth® subsystem with configuration data and the ROM patches</li></ul></li></ul></li></ul>



Number	Feature
12 (continued)	<ul style="list-style-type: none"> <li>○ Provides a transport pipe to send/receive HCI commands/events between the upper stack and the lower layer UART driver, that is, the Bluetooth-IF IDI Driver</li> <li>— Rfkill               <ul style="list-style-type: none"> <li>○ Part of BusyBox* v1.23.2</li> <li>○ Tool for enabling and disabling wireless devices</li> </ul> </li> <li>— Evtest               <ul style="list-style-type: none"> <li>○ Monitors an input device like keyboard or mouse, and display the events the device generates</li> </ul> </li> <li>— Obexctl               <ul style="list-style-type: none"> <li>○ Interactive test application for Obex OPP testing</li> <li>○ Get and Push files from/to remote Bluetooth® devices using OPP</li> </ul> </li> <li>— BT A2DP sink role supported</li> </ul>
13	<p><b>Comms: Wi-Fi*</b></p> <ul style="list-style-type: none"> <li>• This is a release for Comms Wi-Fi enabling through software implementation for Wi-Fi communication. Ports IWL drivers as recipes and make the drivers compatible to the Linux* OS.</li> <li>• Station Mode:               <ul style="list-style-type: none"> <li>— Wi-Fi Station as a device                   <ul style="list-style-type: none"> <li>○ The device can scan and connect to nearby wireless access points with proper login credentials</li> </ul> </li> <li>— Band: 2.4 GHz ISM</li> <li>— Operational modes: 802.11 b/g/n</li> <li>— Security modes: WEP, WPA, WPA2 (TKIP, CCMP)</li> </ul> </li> <li>• HostAP Mode:               <ul style="list-style-type: none"> <li>— HostAP mode provides the wireless network service. The modes can be configured in "/etc/hostapd.conf"</li> <li>— Supports 2.4 GHz ISM band for softAP</li> <li>— Operational modes: 802.11 b/g/n</li> <li>— Hostapd is a user space softAP capable of turning a normal WLAN interface into access points and authentication servers</li> </ul> </li> <li>• IWL Linux NVM Partition:               <ul style="list-style-type: none"> <li>— NVM partition contains nvmData (the WLAN-provisioned data)</li> </ul> </li> </ul>





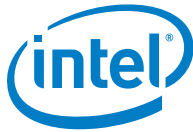
Number	Feature
14	<p><b>COMMS: GNSS and A-GNSS</b></p> <p>This is a release for Comms Global Navigation Satellite System (GNSS) enabling through software implementation for autonomous and assisted fix.</p> <p><b>Packages</b></p> <ul style="list-style-type: none"><li>• Kernel driver – handles the IDI interface to communicate with the GNSS chip</li><li>• LBSD – user space daemon, performs the GNSS calculation and control.</li><li>• Sanity – application is allowed to trigger a fix request.</li><li>• STELP – test application to validate GNSS test cases and scenarios</li><li>• Constellations: GPS, GLONASS.</li><li>• Fix methods: single shot, periodic.</li><li>• Feature Support</li><li>• Assisted GPS</li><li>• Assisted GLONASS as part of Assisted GNSS (AGPS + AGLONASS)</li><li>• SUPL SET Based and SET Assisted</li><li>• SUPLv1.0 and v2.0</li><li>• SUPL Cell ID, Enhanced Cell ID-based Position Fix</li><li>• SUPL over WLAN and CPLANE (2G and 3G)</li></ul>
15	<p><b>systemd</b></p> <ul style="list-style-type: none"><li>• The Linux OS system and service manager enabled by default.</li><li>• SUPL over WLA and CPLANE (2G and 3G)</li></ul>



Number	Feature										
16	<p><b>OTA</b></p> <ul style="list-style-type: none"> <li>Firmware Update           <ul style="list-style-type: none"> <li>A selectable firmware update list from PSI-flash, SLB, MobileVisor* and config, Secure VM, Boot image and Recovery image.</li> <li>The firmware update list is limited by the 80 MB size of fw-update partition.</li> </ul> </li> <li>Software Update           <ul style="list-style-type: none"> <li>Linux* Integrity Measurement Architecture (IMA)-protected file-based Linux OS root-fs update.</li> <li>Files generated for software update are signed with IMA private key offline in build machine.</li> </ul> </li> <li>The OTA update is triggered from the Linux OS under normal mode and the entire OTA update is performed under Linux OS recovery mode.</li> </ul> <p><b>NOTE:</b> As the OTA package is passed from normal mode to recovery mode over the 100 MB-cache partition, the total size of the OTA package (for both software update and firmware update) should not exceed 100 MB.</p>										
17	<p><b>I<sup>2</sup>S*</b></p> <ul style="list-style-type: none"> <li>Supports both master and slave mode.</li> <li>Supports audio playback and recording.</li> <li>Available audio path</li> </ul> <table border="1"> <tr> <td>Record</td><td>Playback</td></tr> <tr> <td>I<sup>2</sup>S</td><td>I<sup>2</sup>S</td></tr> <tr> <td>AFE</td><td>AFE</td></tr> <tr> <td>I<sup>2</sup>S</td><td>AFE</td></tr> <tr> <td>AFE</td><td>I<sup>2</sup>S</td></tr> </table>	Record	Playback	I <sup>2</sup> S	I <sup>2</sup> S	AFE	AFE	I <sup>2</sup> S	AFE	AFE	I <sup>2</sup> S
Record	Playback										
I <sup>2</sup> S	I <sup>2</sup> S										
AFE	AFE										
I <sup>2</sup> S	AFE										
AFE	I <sup>2</sup> S										
18	<p><b>Other Features</b></p> <ul style="list-style-type: none"> <li>Linux OS <b>S0</b> and <b>S3</b> support on PRH-based V<sub>MM</sub> power handler</li> <li>Device Tree for Fab-X, Fab-A, and Fab-B</li> <li>Added capability to switch from usif1-spi/usif2-uart to usif1-uart/usif2-spi (during compile time)</li> <li>USB Device Mode and Host Mode detection (under batteryless configuration)</li> </ul>										



Number	Feature
	<ul style="list-style-type: none"><li>• Integration of kernel v4.3 backported to the Bluetooth® subsystem. Supports Bluetooth® low energy (LE) technology, Generic Attribute Profile (GATT), HID over GATT (HOGP), and Health Thermometer Profile (HTP) profiles.</li><li>• Removed SecureVM, VMM, ModemVM, PSI/SLB, and SocLib components.</li><li>• System shutdown via command line</li><li>• Yocto Project* v2.0.2 (upgrade and security CVEs)</li><li>• Secure Boot Re-signing Tools.</li><li>• Build System supports eMMC and Memory XML layout change for OSAG binaries</li><li>• SoCWatch</li><li>• Connman integrated</li><li>• Battery Charger capability (only available in configuration of Wilbur Bay with graphics and without TEE)</li><li>• Pulse Width Modulation (only available in Wilbur Bay)</li><li>• TEE (Trusted Execution Environment)</li></ul>



Number	Feature
19	<p><b>Graphics and Multimedia Features</b></p> <ul style="list-style-type: none"> <li>• 2D/3D graphics rendering using ARM Mali* DDK* based on the Wayland* environment and X Window System* (X11) environment backend.</li> <li>• Direct Rendering Manager/Kernel Mode Setting (DRM/KMS) kernel driver support for display.</li> <li>• OpenGL ES 1.1, 2.0</li> <li>• OpenVG 1.1</li> <li>• X Window System environment and Wayland environment windowing system</li> <li>• OpenVG support limited to Wayland environment backend</li> <li>• Touchscreen support enabled</li> <li>• System level power management (Suspend/resume for Graphics IP)</li> <li>• MIPI* Display support at 1080p @60</li> <li>• LVDS Display support at 800x480@60</li> <li>• Video decoding using Rockchip* Video Processing Unit (VPU) <ul style="list-style-type: none"> <li>— H.263: Base profile 480p @30fps</li> <li>— H.264: Main Profile up to level 4.1, 1080p @ 30fps</li> <li>— H.265: Main profile up to level 5.1, 1080p @ 30fps</li> <li>— VP8: 1080p@ 30fps</li> <li>— MPEG-2/4: Simple profile up to level 6 and Advanced Simple profile up-to level 5.</li> </ul> </li> <li>• Video post processing using Rockchip Raster Graphics Accelerator (RGA) up to 1920x1080 @30fps <ul style="list-style-type: none"> <li>— Color space conversion from NV12, YUV420 to BGRA, BGRx</li> <li>— Horizontal and vertical video flipping</li> <li>— Video upscaling and downscaling from 240p to 1080p <ul style="list-style-type: none"> <li>• Video encoding using Rockchip Video Processing Unit (VPU)</li> </ul> </li> <li>— H.264 encoding up to high profile, level 4.1 1080p@30fps.</li> </ul> </li> <li>• Hardware accelerated video playback using GStreamer multimedia framework; rendering through Wayland and X Window System environments.</li> </ul>



## 1.9 New Features

- Fix for new attacks on WLAN WPA supplicants, which broken Wi-Fi handshakes.

## 1.10 Limitations

None for this release.

## 1.11 Known Issues

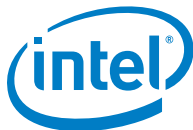
**Table 9. Known Issues**

Defect ID	Description	Status
1504243897	port111 is open and unfiltered.	Would not be fixed
1504297091	Power button does not turn off display.	Would not be fixed
1504300664	USB keyboard issue when connected to USB hub to DUT.	Would not be fixed
1504325487	Incorrect /sys/kernel/debug/mid_pmu_states for accelerometer, gyroscope, magnetometer sensor.	Would not be fixed
1504312902	[Telit] Cross talk (channel leak) during audio playback on 3.5 mm jack.	Open
1804305651	[KPI][POWER] No S3 state on Yocto with SoFIA 3GR (Refer Appendix for workaround).	Closed with workaround
1504609980	Porting SF_LTE good known fixes to SF_3GR	Open

## 1.12 Fixed Issues

**Table 10. Fixed Issues – Others**

None



**Table 11. Fixed Issues - Communications**

Defect ID	Title	Comp	Description	Resolution
<b>1604518763</b>	[SF3GR] Wi-Fi Protected Access II (WPA2) handshake traffic can be manipulated to induce nonce and session key reuse	Wi-Fi	Wi-Fi Protected Access II (WPA2) handshake traffic can be manipulated to induce nonce and session key reuse, resulting in key reinstallation by a wireless access point (AP) or client. An attacker within range of an affected AP and client may leverage these vulnerabilities to conduct attacks that are dependent on the data confidentiality protocols being used. Attacks may include arbitrary packet decryption and injection, TCP connection hijacking, HTTP content injection, or the replay of unicast and group addressed frames.	Refer to <a href="#">INTEL-SA-00101</a> for more information

## 1.13 Where to Find the Release

This release is available for download from the Intel RDC website by searching for *Intel Atom® Processor C3200 Series*.

§



## 2.0 *Setting up Build Environments*

---

### 2.1 **Setting up the Host Machine**

The user is expected to build the Yocto Project\*-based image from the source code, following the instructions in [Section 3.0, Compiling the Build](#). This section contains guidelines to set up a build host machine.

An Ubuntu\* 64-bit OS version 14.04 is recommended with the following hardware requirements.

- Intel® Core™ i7 Processors
- 8 GB RAM
- 500 GB storage
- Internet connection

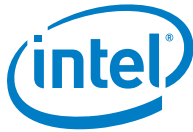
**Note:** Other Linux\* distributions or other Ubuntu\* OS versions are not verified.

#### 2.1.1 **Proxy Server Configuration for Internet Access**

If the build machine is behind a corporate network, it needs to access the Internet via a proxy server. Skip this section if the build machine is not behind any proxy server.

Add the following proxy server settings to `/etc/environment` and `~/.bashrc`:

```
export HTTP_PROXY=<proxy server IP or DNS>:<http port>
export HTTPS_PROXY=<proxy server IP or DNS>:<https port>
export FTP_PROXY=<proxy server IP or DNS>:<ftp port>
export SOCKS_SERVER=<proxy server IP or DNS>:<socks port>
```



## 2.1.2 Installing Additional Packages

1. If the host system is behind a proxy network, preconfigure the `/etc/apt/apt.conf` with proxy server by adding the following lines in `/etc/apt/apt.conf`

```
Acquire::http::proxy "http://<proxy server IP or DNS>:<http
port>/" ;
Acquire::https::proxy "http://<proxy server IP or
DNS>:<https port>/" ;
Acquire::ftp::proxy "http://<proxy server IP or DNS>:<ftp
port>/" ;
Acquire::socks::proxy "socks://<proxy server IP or
DNS>:<socks port>/" ;
```

2. Install these packages in Ubuntu\* OS version 14.04:

```
$ sudo apt-get install -y gawk wget git-core diffstat unzip
texinfo gcc-multilib build-essential chrpath socat
$ sudo apt-get install -y libsdl1.2-dev xterm
$ sudo apt-get install -y make xsltproc docbook-utils fop
dblatex xmlto
$ sudo apt-get install -y autoconf automake libtool
libglib2.0-dev
```

3. The following software packages are needed to build the **mvconfig** tool in `meta-intel-sofia-sw/meta-intel-sofia-mobilevisor`:

```
$ sudo apt-get install -y lib32stdc++6 libxml2:i386
libxml2-dev:i386
```





### 2.1.3 Setting Up git Configuration for Open Source GIT Repository

1. Set the git configuration by creating a **gitconfig** file and appending the following lines in ~/.gitconfig:

```
[user]
email = your.name@yourcompany.com
name = "YOUR NAME"
[sendemail]
smtpserver = smtp.yourcompany.com
signedoffcc = false
suppresscc = all
chainreplyto = false
assume8bitEncoding = utf-8
from = "YOUR NAME <your.name@yourcompany.com>"
confirm = always

[color]
diff = auto
ui = auto
interactive = auto
grep = always

[color "grep"]
match = red

[alias]
co = checkout
br = branch
ci = commit
st = status
ol = log -oneline

[core]
editor = vi    # Note: you can choose other editor like
gedit
gitproxy = /home/<user>/common/bin/gitproxy
```

**Note:** Accessing the **git** repository over a proxy server requires the **gitproxy** script. If the build machine is not behind a proxy server, remove the following line:

```
gitproxy = /home/<user>/common/bin/gitproxy
```

2. Intel recommends the following content for /home/<user>/common/bin/gitproxy:

```
#!/bin/sh
# Choose the site proxy server accordingly
exec socat stdio SOCKS:<proxy server IP or DNS>:$1:$2
```

3. Make the script executable:



```
$ chmod +x /home/<user>/common/bin/gitproxy
```

## 2.2 Installing the Build System

To set up the build environment in the system, a script is included in the package listed in [Table 3](#). Run the `setup-build-env-ext` script that gets the Free and Open Source Software (FOSS) software over the Internet, unpacks packed tarballs, and combines all ingredients in the correct locations.

**Note:** `setup-build-env-ext` gets the `repo` tool and puts it at the following location: `~/common/bin`. The user can change this location.

**Note:** `setup-build-env-ext` extends the environment variable `PATH` to include the `~/common/bin` path that contains the `repo` tool. The user needs to add this `PATH` manually, to `~/ .bashrc`.

Run the following procedure to set up the build system.

1. Assuming that the release tarball is copied to the home directory, run the following commands.

```
$ mkdir ~/repo
$ mv sofia3gr_<version>.tar.bz2 ~/repo
$ cd ~/repo
$ tar -xvf sofia3gr_<version>.tar.bz2
```

2. After the tar files are extracted, change the directory to the release directory.

```
$cd sofia3gr_<version>
```

3. Run the following command to get the repository.

```
$source setup-build-env-ext repo -b master --linux
```

**Note:** After executing the preceding steps, follow [Section 3.1](#) to compile the build.

## 3.0 *Compiling the Build*

---

### 3.1 **Running the Setup Script**

To ease the integration of poky\*, **meta-intel**, **meta-intel-sofia-sw**, **meta-intel-sofia** and **meta-intel-sofia-mobilevisor** and eliminate steps to correctly set up **local.conf** and **bblayers.conf**, all customizations are included within the **setup-build-env-ext** script. Intel recommends studying the script to understand how the script parses the sample templates (**local.conf.sample** and **bblayers.conf.sample**) defined under `meta-intel-sofia-sw/conf`.

The following commands are required to create the `build/conf` configuration files and start the BitBake operation to build the complete software ingredients for Intel Atom® processor C3200 series. Use either the X Window System\* environment or the Wayland environment as the default windowing system. Follow [Section 3.2](#) or [Section 3.3](#), depending on the desired selection.

```
$ cd ~/repo/sofia3gr_<version>
$ source setup-build-env-ext build -m sofia-3gr
```

### 3.2 **Configure the BSP to Enable Different Platform and Display Support**

Enable the platform and display support from [Section 3.2](#) and select the desired windowing system from [Section 3.3](#) or [Section 3.4](#).

This section provides instructions to configure the BSP to support different platform and display configurations. The available combination of platform and display is mentioned in [Table 7](#). The following are the instructions:

1. Change the directory to “build-sofia-3gr”

```
$ cd ~/repo/sofia3gr_<version>
```

```
$ cd build-sofia-3gr/conf
```

2. The following snippet shows the configuration in the `local.conf` file. Edit the `local.conf` file by commenting out the default and uncommenting the desired settings. The default configuration is for Telit\* HE922-3G R hardware shown as follows:

```
# Please select the desired platform build by commenting out
default
# and uncommenting the desired settings. By default, it is
Telit HE922
# with MIPI panel.
```



```
#
# Malibu Beach (headless)
# =====
#LUNCH_PLATFORM = "sf3gr_crb_h1"
#MACHINE_FEATURES_remove = " trusty"
#KERNEL_NORMAL_DT = "SF_3GR-crb-v1.dtb"
#KERNEL_RECOVERY_DT = "SF_3GR-crb-v1-recovery.dtb"
#MANUFACTURER = "Intel"
#MODEL_NO = "1.0"
#
# Malibu Beach with LVDS panel
# (Same LUNCH-TARGET for Wilbur-Bay but adjusting the
# MACHINE_FEATURE)
# =====
LUNCH_PLATFORM = "sf3gr_crb_v1_lvds"
MACHINE_FEATURES_remove = " trusty"
KERNEL_NORMAL_DT = "SF_3GR-crb-v1-lvds.dtb"
KERNEL_RECOVERY_DT = "SF_3GR-crb-v1-lvds-recovery.dtb"
DISPLAYPANEL_TYPE = "lvds"
MANUFACTURER = "Intel"
MODEL_NO = "1.0"
#
# Malibu Beach with MIPI panel
# =====
#LUNCH_PLATFORM = "sf3gr_crb_v1"
#MACHINE_FEATURES_remove = " trusty"
#KERNEL_NORMAL_DT = "SF_3GR-crb-v1.dtb"
#KERNEL_RECOVERY_DT = "SF_3GR-crb-v1-recovery.dtb"
#DISPLAYPANEL_TYPE = "mipi"
#MANUFACTURER = "Intel"
#MODEL_NO = "1.0"
#
# Telit HE922 with MIPI panel
# (Same LUNCH-TARGET for WE922 but adjusting the
# MACHINE_FEATURE)
# =====
#LUNCH_PLATFORM = "sf3gr_telit_he922"
#MACHINE_FEATURES_remove = " trusty"
```



```
#KERNEL_NORMAL_DT = "SF_3GR-telit-he922.dtb"
#KERNEL_RECOVERY_DT = "SF_3GR-telit-he922-recovery.dtb"
#DISPLAYPANEL_TYPE = "mipi"
#MANUFACTURER = "Telit"
#MODEL_NO = "1.0"
#
# Trusty Malibu Beach with MIPI panel
# =====
# LUNCH_PLATFORM = "sf3gr_crb_v1_trusty"
# MACHINE_FEATURES_append =" trusty"
# KERNEL_NORMAL_DT = "SF_3GR-crb-v1.dtb"
# KERNEL_RECOVERY_DT = "SF_3GR-crb-v1-recovery.dtb"
# DISPLAYPANEL_TYPE = "mipi"
# MANUFACTURER = "Intel"
# MODEL_NO = "1.0"
#
# Trusty Telit HE922 (headless)
# =====
# LUNCH_PLATFORM = "sf3gr_he_hl_trusty"
# MACHINE_FEATURES_append =" trusty"
# KERNEL_NORMAL_DT = "SF_3GR-telit-he922.dtb"
# KERNEL_RECOVERY_DT = "SF_3GR-telit-he922-recovery.dtb"
# MANUFACTURER = "Telit"
# MODEL_NO = "1.0"
#
# Trusty Telit HE922 with MIPI panel
# =====
#LUNCH_PLATFORM = "sf3gr_he_trusty"
#MACHINE_FEATURES_append =" trusty"
#KERNEL_NORMAL_DT = "SF_3GR-telit-he922.dtb"
#KERNEL_RECOVERY_DT = "SF_3GR-telit-he922-recovery.dtb"
#MANUFACTURER = "Telit"
#MODEL_NO = "1.0"
#
#
# Telit HE922 (headless)
# =====
# LUNCH_PLATFORM = "sf3gr_telit_he_hl"
```



```
# MACHINE_FEATURES_remove =" trusty"
# KERNEL_NORMAL_DT = "SF_3GR-telit-he922.dtb"
# KERNEL_RECOVERY_DT = "SF_3GR-telit-he922-recovery.dtb"
# MANUFACTURER = "Telit"
# MODEL_NO = "1.0"
#
# Trusty Malibu Beach with LVDS panel
# (Same LUNCH-TARGET for Wilbur-Bay but adjusting the
MACHINE_FEATURE)
# =====
# LUNCH_PLATFORM = "sf3gr_v1_lv_trusty"
# MACHINE_FEATURES_append =" trusty"
# KERNEL_NORMAL_DT = "SF_3GR-crb-v1-lvds.dtb"
# KERNEL_RECOVERY_DT = "SF_3GR-crb-v1-lvds-recovery.dtb"
# DISPLAYPANEL_TYPE = "lvds"
# MANUFACTURER = "Intel"
# MODEL_NO = "1.0"
#
# Malibu Beach with Battery charger
# =====
#LUNCH_PLATFORM = "sf3gr_crb_v1_bat"
#MACHINE_FEATURES_remove =" trusty"
#MACHINE_FEATURES_append =" battery"
#KERNEL_NORMAL_DT = "SF_3GR-crb-v1.dtb"
#KERNEL_RECOVERY_DT = "SF_3GR-crb-v1-recovery.dtb"
#DISPLAYPANEL_TYPE = "mipi"
#MANUFACTURER = "Intel"
#MODEL_NO = "1.0"
#
#
# Malibu Beach Trusty (headless)
# (Same LUNCH-TARGET for Wilbur-Bay but adjusting the
MACHINE_FEATURE)
# =====
# LUNCH_PLATFORM = "sf3gr_v1_hl_trusty"
# MACHINE_FEATURES_append =" trusty"
# KERNEL_NORMAL_DT = "SF_3GR-crb-v1.dtb"
# KERNEL_RECOVERY_DT = "SF_3GR-crb-v1-recovery.dtb"
```



```
# MANUFACTURER = "Intel"
# MODEL_NO = "1.0"
#
```

**Note:** Ensure there is no space between the first character and the starting of the line after deleting the “#” (to uncomment the desired setting). Unnecessary spacing causes compilation error.

3. For Wilbur Bay or Telit WE922, edit *sofia-3gr.conf* as below:

a) Wilbur Bay: from Step 2, uncomment Malibu Beach lunch target and then:

```
vi <bsp>/poky-sofia/meta-intel-sofia-sw/meta-intel-
sofia/conf/machine/sofia-3gr.conf
```

```
MACHINE_FEATURES_remove = " pci acpi battery usbconsole
socwatch phone gnss bluetooth wifi"
```

```
MACHINE_FEATURES_append = " screen touchscreen
${DISPLAYPANEL_TYPE} odisclient"
```

b) Telit WE922: from Step 2, uncomment Telit lunch target and then:

```
vi <bsp>/poky-sofia/meta-intel-sofia-sw/meta-intel-
sofia/conf/machine/sofia-3gr.conf
```

```
MACHINE_FEATURES_remove = " pci acpi battery usbconsole
socwatch phone gnss"
```

```
MACHINE_FEATURES_append = " screen touchscreen
${DISPLAYPANEL_TYPE} odisclient bluetooth wifi"
```

### 3.3 Enabling Wayland\* Protocol as Default Windowing System

This section describes the method to enable the Wayland\* protocol on the processor. Make the Wayland environment the default windowing system by editing the *local.conf* file.

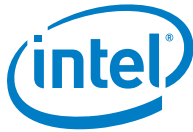
1. Change the directory to “build-sofia-3gr”.

```
$ cd ~/repo/sofia3gr_<version>
```

```
$ cd build-sofia-3gr/conf
```

2. Edit the *local.conf* to enable the Wayland environment.

```
$vim local.conf
```



3. Comment the following lines.

```
#DISTRO_FEATURES_append = " x11"

#DISTRO_FEATURES_remove = "wayland"
```

**Note:** *pokysofia.conf* chooses Wayland as the default windowing system.

4. Save and exit.
5. Exit the "build-sofia-3gr/conf" directory and start BitBake to build the binaries.

```
$ bitbake sofia-sw
```

After the BitBake execution is complete, the flash binaries are available in `build-sofia-3gr/tmp-jethro-sofia-3gr/deploy/fls-binaries/sofia-3gr`

6. The following `.fls` files are the flash binary files required to flash the SoFIA 3G R device:

```
psi_flash_signed.flc,slb_signed.flc,mobilevisor_signed.flc,
mvconfig_smp_profiling_signed.flc,boot-signed.flc,
mvconfig_smp.flc, recovery_signed.flc,secvm_signed.flc,
splash_img_signed.flc,system_signed.flc,ucode_patch_signed.flc,
cache_signed.flc ,vrl_signed.flc
```





The preceding flash files are also packaged in a zip file located in the `sofia_3gr_<version>` folder. The zip file format is `sofia-3gr-flashfiles-<date-time-stamp>.zip`. The zip file can be used directly to program the target machine by using the Intel® Platform Flash Tool (Intel® PFT) explained in [Section 4.0 Flashing Binaries on Target Platform](#).

### 3.3.1 Launching Wayland Protocol on Target System

This section describes the method to start the Wayland environment on the target after the binaries are flashed onto it. Refer to [Section 4.0](#) for the image flashing method.

1. Connect the Serial UART interface and log in as root. (Refer to [Section 1.4 Reference Documents](#) and look for the *Intel Atom® X3-C3200RK SoFIA Yocto Project\* - Getting Started Guide*, Section 9.0) for detailed instructions on connecting the serial interface)
2. Once logged in, issue the following commands to initiate the Wayland environment.

```
$ XDG_RUNTIME_DIR=/tmp
$ openvt -s -- weston --backend=drm-backend.so --idle-
time=86400
```

### 3.3.2 Integrating and Testing Weston\* Applications

To integrate the test applications into the build, edit the `local.conf` file in the host.

1. Change the directory to “build-sofia-3gr”  

```
$ cd ~/repo/sofia3gr_<version>
$ cd build-sofia-3gr/conf
```
2. Edit the `local.conf` file to enable the Wayland environment.

```
$vim local.conf
```

3. Comment the following lines.

```
#DISTRO_FEATURES_append = " x11"
#DISTRO_FEATURES_remove = "wayland"
```

**Note:** `pokysofia.conf` chooses the Wayland environment as the default windowing system.

4. Insert the following line at the end of the file.

```
IMAGE_INSTALL_append = `` weston-examples``
```

5. Save the file and exit.



6. Exit the "build-sofia-3gr/conf" directory and start BitBake to build the binaries.

```
$ bitbake sofia-sw
```

After the image is built, follow [Section 3.3.1](#) to flash and start the Wayland environment.

After the Wayland environment is launched in the target, test the following examples:

```
weston, weston-calibrator, weston-clickdot, weston-cliptest,
weston-dnd, weston-editor, weston-eventdemo, weston-flower,
weston-fullscreen, weston-image, weston-info, weston-launch,
weston-multi-resource, weston-presentation-shm, weston-resizor,
weston-scaler, weston-simple-damage, weston-simple-egl, weston-
simple-shm, weston-simple-touch, weston-smoke, weston-stacking,
weston-surfaces, weston-terminal and weston-transformed
```

## 3.4 Enabling X Window System\* (X11) as Default Windowing System

This section describes the method to enable the X Window System\* on SoFIA 3G R. By default, `pokysofia.conf` chooses the Wayland environment as the default windowing system. Edit the `local.conf` file to change the environment to X Window System.

1. Change the directory to "build-sofia-3gr"

```
$ cd ~/repo/sofia3gr_<version>
```

```
$ cd build-sofia-3gr/conf
```

2. Edit the `local.conf` file to enable X-Window System.

```
$vim local.conf
```

3. Uncomment the following lines.

```
DISTRO_FEATURES_append = " x11"
```

```
DISTRO_FEATURES_remove = "wayland"
```

4. Save and exit.

5. Exit the "build-sofia-3gr/conf" directory and Start BitBake to build the binaries.

```
$ bitbake sofia-sw
```

After the BitBake execution is complete, the flash binaries are available in `build-sofia-3gr/tmp-jethro-sofia-3gr/deploy/fls-binaries/sofia-3gr`

6. The following .fls files are the flash binary files required to flash the SoFIA 3G R device:



```
psi_flash_signed.flc,slb_signed.flc,mobilevisor_signed.flc,  
mvconfig_smp_profiling_signed.flc,boot-signed.flc,  
mvconfig_smp.flc, recovery_signed.flc,secvm_signed.flc,  
splash_img_signed.flc,system_signed.flc,ucode_patch_signed.flc,  
cache_signed.flc ,vrl_signed.flc
```

The preceding flash files are also packaged in a zip file located in the `sofia_3gr_<version>` folder. The zip file format is `sofia-3gr-flashfiles-<date-time-stamp>.zip`. This zip file can be used directly to program on-top target machine by using the Intel® PFT explained in [Section 4.0 Flashing Binaries on Target Platform](#).

### 3.4.1 Testing X Window System (X11) on SoFIA 3G R

This section describes the method to start X11 on the target after the binaries are flashed onto it. Refer to [Section 4.0](#) for the image flashing method.

1. Connect the Serial UART interface and log in as root. (Refer to [Section 1.4 Reference Documents](#), *Intel Atom® X3-C3200RK SoFIA Yocto Project\* - Getting Started Guide*, Section 9.0 for detailed instructions on connecting the serial interface.)
2. Once logged in, issue the following command to start X11.

```
$ xorg -retro &
```

3. Run the sample xeyes test.

```
$ export DISPLAY=:0.0 ; xeyes
```

## 3.5 Optimizing `conf/local.conf`

The duration of the BitBake process can be reduced considerably by using a common path for downloads and `sstate-cache` folders.

During the build process, the sources of FOSS projects hosted in the Internet are downloaded into the `build/downloads` folder. At the same time, BitBake can populate the `tmp/work` build folders with pre-built artifacts that are cached under the `build/sstate-cache`. Intel recommends setting up a common path for both downloads and the `sstate-cache` folders to shorten the BitBake build process over time.

To set up a common path, define the path for `TOPDIR` shown as follows in `build-sofia-3gr/conf/local.conf`. Create the `yocto-build` before starting the BitBake build process.

```
$ mkdir ~/yocto-build
```

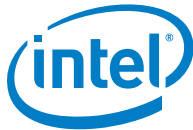


Figure 1. TOPDIR Usage in conf/local.conf

```
# Where to place downloads
#
# During a first build the system will download many different source code tarballs
# from various upstream projects. This can take a while, particularly if your network
# connection is slow. These are all stored in DL_DIR. When wiping and rebuilding you
# can preserve this directory to speed up this part of subsequent builds. This directory
# is safe to share between multiple builds on the same machine too.
#
# The default is a downloads directory under TOPDIR which is the build directory.
TOPDIR = "/home/user/yocto-build"
DL_DIR ?= "${TOPDIR}/downloads"
#
# Where to place shared-state files
#
# BitBake has the capability to accelerate builds based on previously built output.
# This is done using "shared state" files which can be thought of as cache objects
# and this option determines where those files are placed.
#
# You can wipe out TMPDIR leaving this directory intact and the build would regenerate
# from these files if no changes were made to the configuration. If changes were made
# to the configuration, only shared state files where the state was still valid would
# be used (done using checksums).
#
# The default is a sstate-cache directory under TOPDIR.
SSTATE_DIR ?= "${TOPDIR}/sstate-cache"
#
# Where to place the build output
#
# This option specifies where the bulk of the building work should be done and
# where BitBake should place its temporary files and output. Keep in mind that
# this includes the extraction and compilation of many applications and the toolchain
# which can use Gigabytes of hard disk space.
#
# The default is a tmp directory under TOPDIR.
TMPDIR = "${TOPDIR}/tmp-${BRANCH}-${MACHINE}"
```

**Note:** When the preceding configuration is complete,  
~/repo/sofia3gr\_<version>/build-sofia-3gr contains just the conf folder.  
Unnecessary spacing causes compilation error.



## 4.0 *Flashing Binaries on Target Platform*

---

The Intel® Platform Flash Tool (Intel® PFT) flashes the binaries built from the build system into the eMMC\* of the target platform. The tool is available for the following OSes:

- Windows\* OS
- Linux\* OS
- Mac OS\* (64-bit)

Refer to [Section 1.4, Reference Documents](#) for the IBP number to download the Intel® PFT.

**Note:** For this release, flash the Android\* OS to the system before flashing the Yocto Project\*-based BSP, to provide NVM data to the system. Also ensure that the modem version used is the same with the modem version on the Android\* OS (The Android\* OS version that is compatible is the Android\* release for SoFIA in ultramobile-kits, 20161125-2016\_ww48 release.).

**Note:** The following flash procedure uses the Intel® Platform Flash Tool (Intel® PFT) installed in the Windows OS.

**Note:** Execute the following steps before the steps in [Section 5.1](#) to have a successful image boot.

1. When flashing the Android\* image, to select the “SMP FLS only” flashing option, as highlighted in [Figure 2](#).

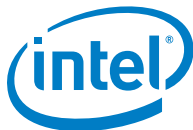
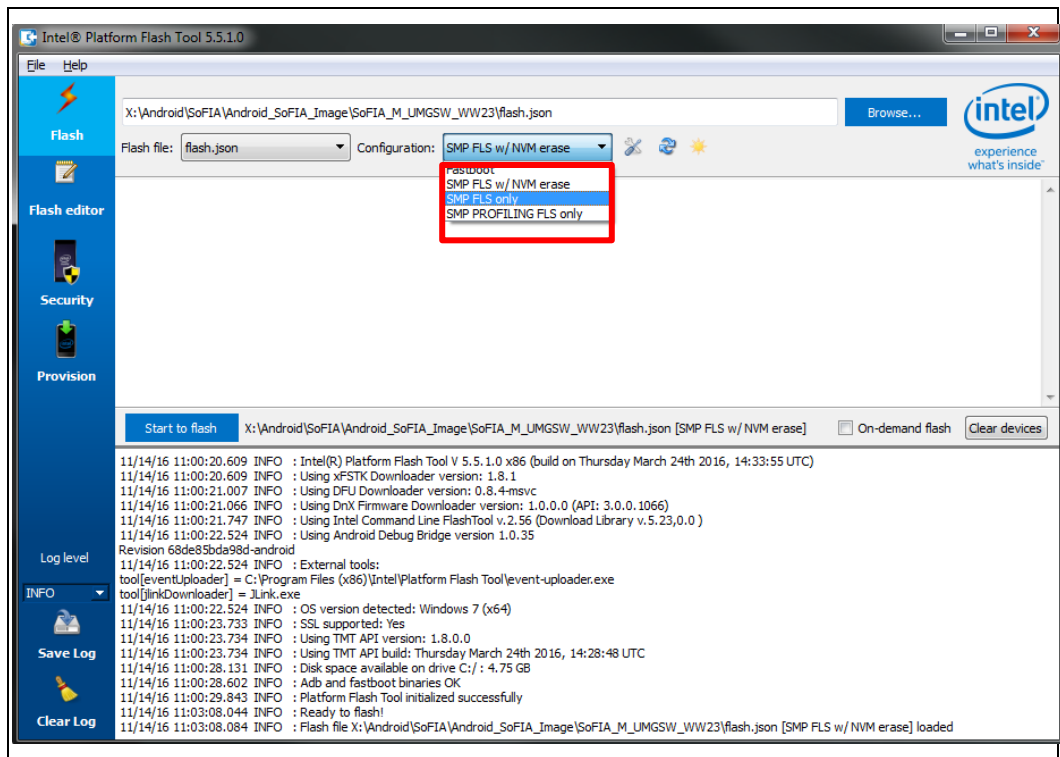


Figure 2. Flashing Android\* OS with “SMP FLS only” Flashing Option



2. Upon completing the flash, boot the system with the Android\* image before flashing the Yocto Linux\* OS onto the system.
3. If needed, enter the `p test` mode for the necessary configuration. Proceed to shut down the system that is running on Android\* OS.

For detailed instructions on flashing the Android\* OS, refer to [Section 1.4 Reference Documents](#) and look for *Android\* OS with Board Support Package for Intel Atom® x3-C3200RK Processor (Formerly SoFIA) Release Notes*.

4. Set the flashing option to “**SMP FLS only**”, as shown in Figure 2, then proceed to [Section 4.1](#).

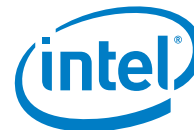
## 4.1 Flashing Instructions with the Intel® Platform Flash Tool (Intel® PFT) Using Windows\* Host

The following are instructions to flash the compiled binary to the target device. The Malibu Beach board is used as a sample hardware.

Refer to [Section 1.4 Reference Documents](#) for the IBP number to download the suggested Intel® PFT.

Unzip the downloaded package to any location on a machine running Windows\* OS.

The following are steps to install the Intel PFT from the package:

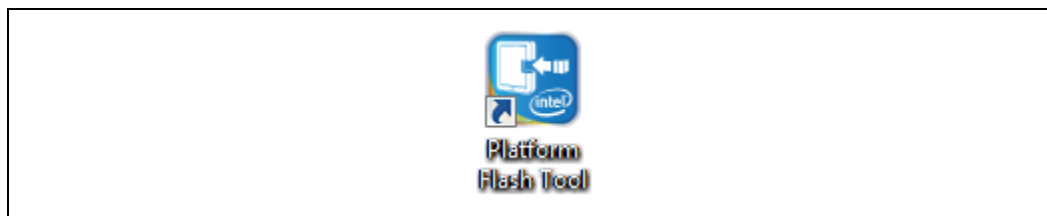


1. Browse to the location where the package is stored and unzip the **PlatformFlashTool\_5.5.1.0\_win32.zip** file.
2. Execute the unzipped **PlatformFlashTool\_5.5.1.0\_win32.exe** (right-click on the .exe file, click **Run As Administrator** to install Intel® PFT and drivers.)
3. Follow the instructions on the interactive installer to complete the installation.

**Note:** The executable file installs the USB drivers required for the Intel PFT automatically on the machine running Windows\* OS.

When the installation is complete, an Intel PFT icon is created on the desktop, as shown in [Figure 3](#).

**Figure 3. Intel® Platform Flash Tool (Intel® PFT) Desktop Icon**



4. Double-click the icon to flash the target platform.
5. Note the tool version on the top left portion of the Intel PFT interactive window, as highlighted in [Figure 4](#). Verify that the Intel PFT version is 5.5.1.0 (recommended version).

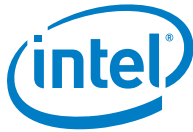
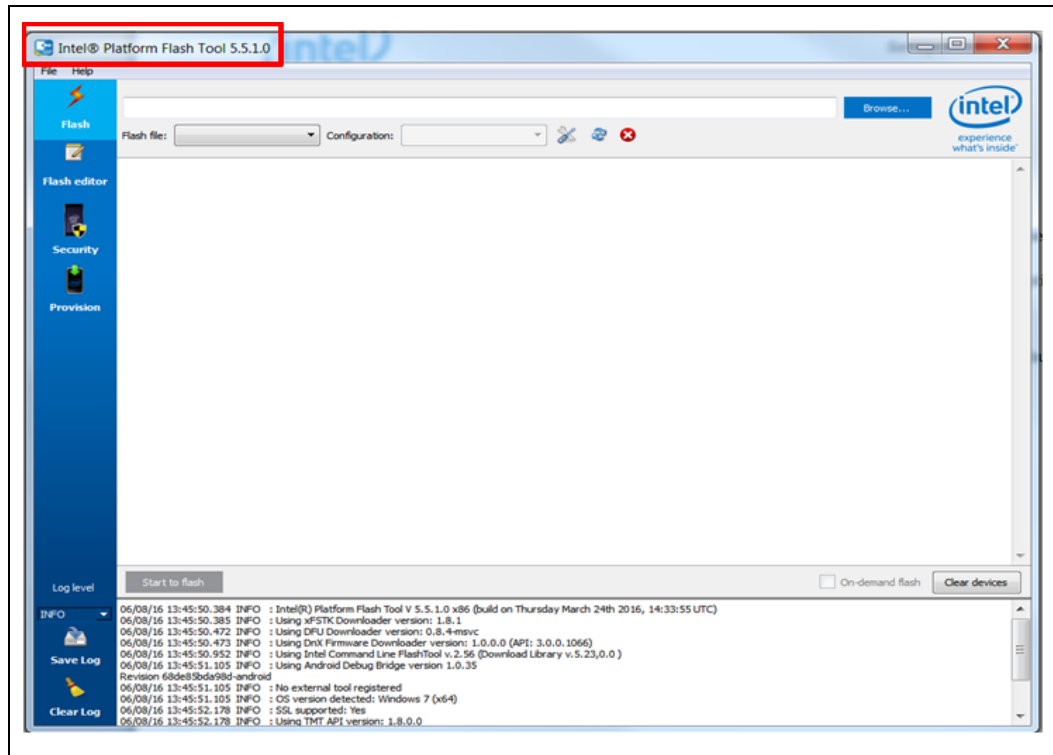


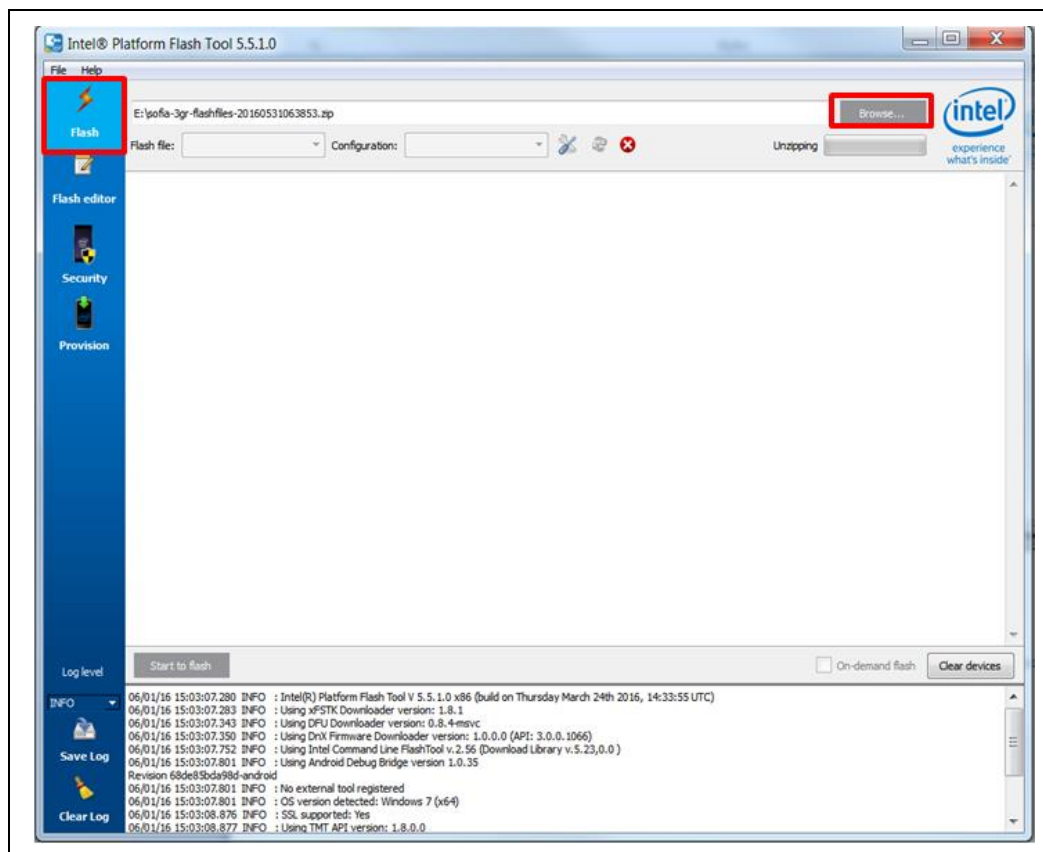
Figure 4. Intel® Platform Flash Tool (Intel® PFT) Window



6. Ensure that the Flash screen option is selected, as shown in [Figure 5](#).
7. Browse to the fls zip file (example: `sofia-3gr-flashfiles.zip`) using the **Browse** button. The tool automatically starts to unzip the fls package.



**Figure 5. Browse to the fls Zip File**

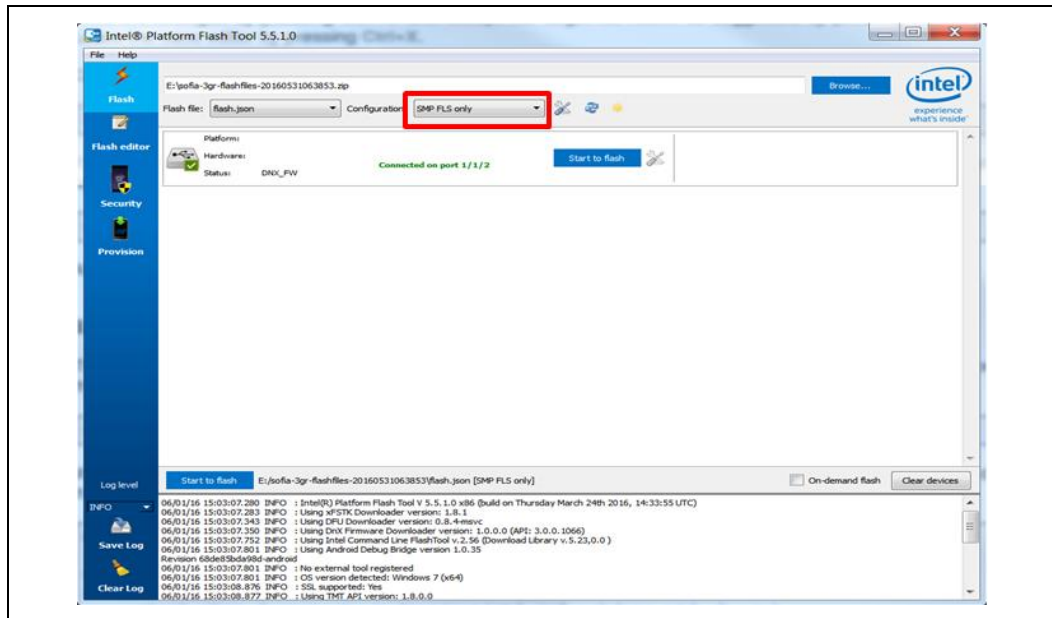


8. Configure the image erase calibration on the Configuration drop-down menu available on the interactive screen, as seen in Figure 6. Ensure that the SMP FLS only configuration option is selected before flashing.

**Note:** The SMP FLS only configuration conditionally erases the NVM data before the FLS binary flashing. By default, the configuration option is on SMP FLS w/NVM erase, where this configuration option erases all NVM data before the FLS binary flashing.



Figure 6. Flashing with “SMP FLS Only”



9. After configuring the Intel PFT, proceed to connect the target platform to any of the available USB ports on the machine running on Window\* OS, using a microUSB cable.
10. Power on the target device. When the target platform boots, the device is detected and listed on the Intel PFT interactive window, as shown in Figure 6.
11. Proceed to flash the target platform by clicking the Start to flash tab on the device listing.

**Note:** The target platform flashes the binary only with the DNX\_FW mode. If the system is out of the DNX\_FW status and the **Start to flash** tab is disabled, reset the target platform.

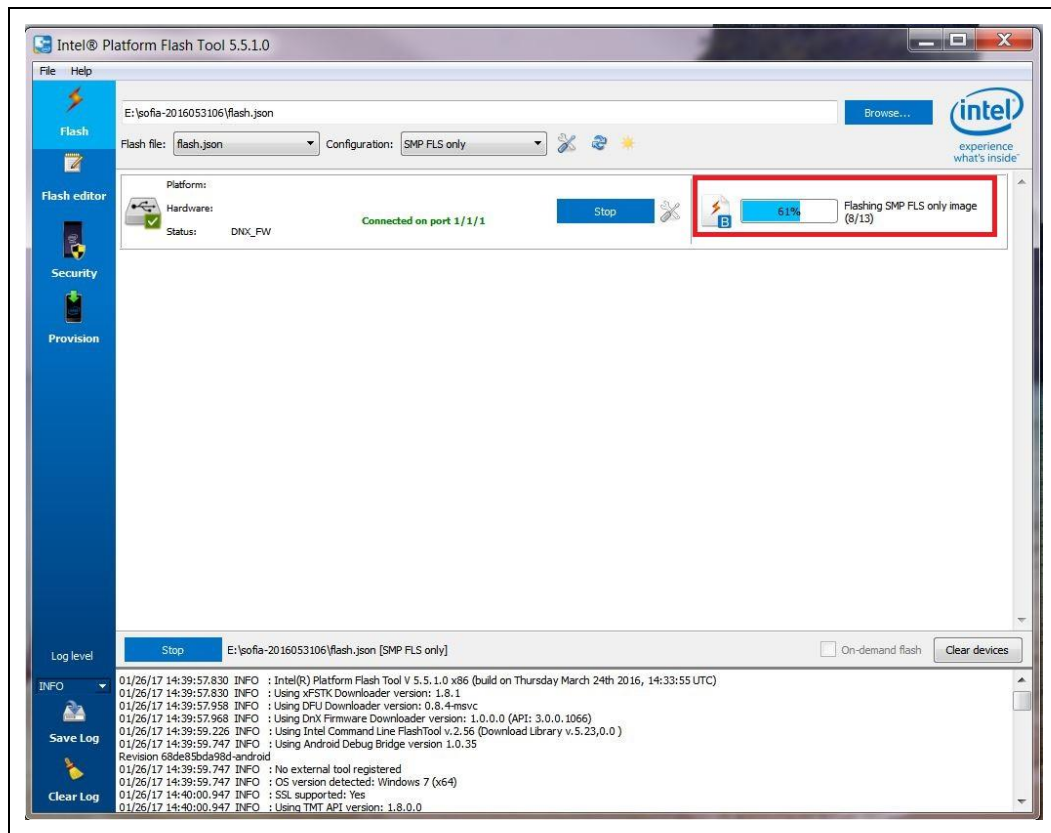
12. Notice that the system starts to flash the .fls binary to the target platform.

**Note:** If the progress bar doesn't respond after the flashing is started, restart the target platform, the flashing sequence will pick up and start flashing.

13. On flash completion, a successful flash message can be seen on the device listing, as highlighted in [Figure 8](#).



Figure 7. Download Progress Screen



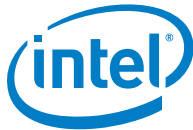
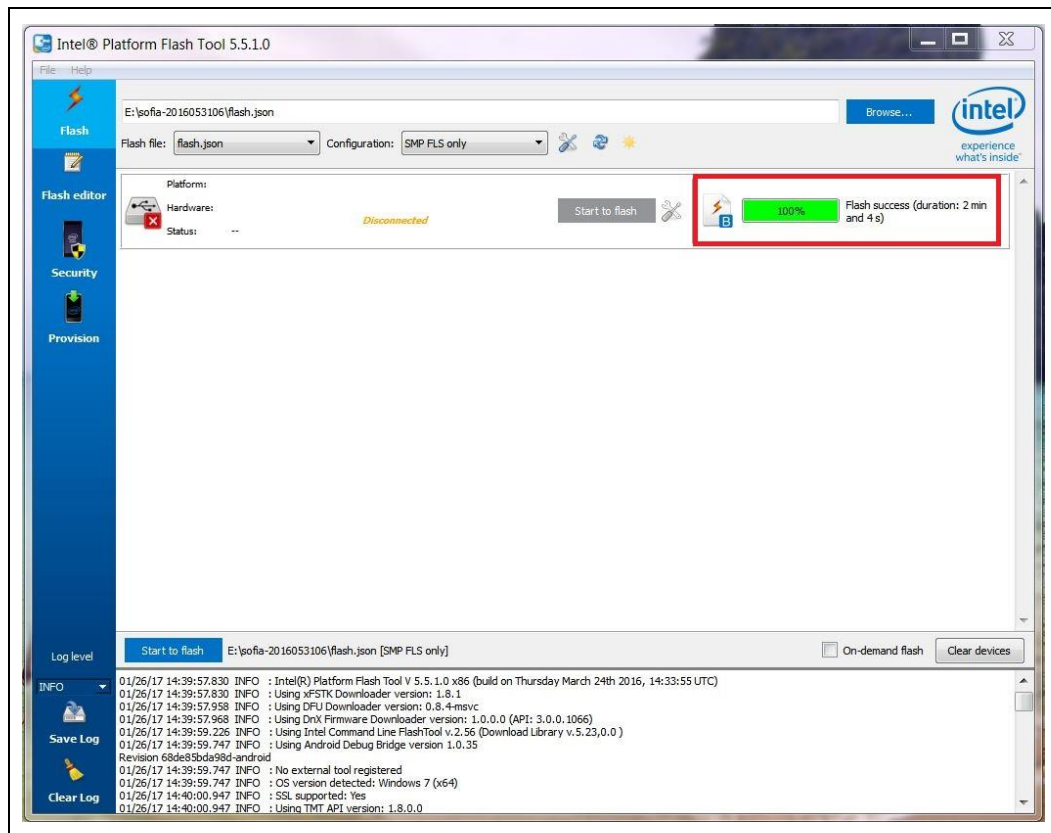
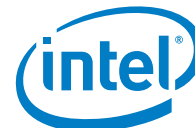


Figure 8. Download Success Screen





14. Before powering up the target platform, connect the USB-to-UART cable between the host machine and target machine. Intel recommends using the PuTTY program for the serial terminal connection. The UART baud-rate used is 115200.
15. Press the **power** button for a few seconds and then release. The target platform starts booting and the splash screen image is displayed on the screen.

As the target platform boots up, a log is displayed on the PuTTY program. When the boot completes, the login name is root. There is no password by default.

## 4.2 Flash Binary Packaging Script

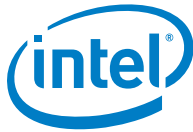
SoFIA OE-class build-system by default always keeps copies of flash binaries built earlier. New flash binaries are tagged with date and time of the build. Symbolic links are always pointing to the latest flash binary available on the build folder. For convenience of extracting flash binaries to be programmed by the Intel PFT onto the Intel Atom® x3-C3200RK/C3230RK board, users can modify the path definition of IN and OUT, to the build path. Make it executable using this command: `chmod +x fls-pkg-gen.sh`

```
#!/bin/bash
IN=/home/<username>/sofia3gr_<version>/build-sofia-3gr/tmp-
jethro-sofia-3gr/deploy/fls-binaries/sofia-3gr
OUT=/home/<username>/sofia-3gr-fls

curdir=$(pwd)
cd $IN
if [ -d fls ]; then
    rm -rf fls
fi
mkdir fls

cp $(find . -type l) fls/
if [ -d $OUT ]; then
    rm -rf $OUT
fi
mv fls $OUT
cd $curdir
```

§



## 5.0 Video Playback Using GStreamer\* Framework

---

The video playback is enabled through the GStreamer\* multimedia framework. The framework consists of open source and proprietary plug-ins from Intel. The rendering can be done through X Window System\* (X11) (ximagesink) and Wayland\* (waylandsink) environments.

### 1. For a simple playback

```
gst-launch-1.0 filesrc location=<file path> ! decodebin ! videorga!
ximagesink
```

```
gst-launch-1.0 filesrc location=<file path> ! decodebin ! videorga!
waylandsink
```

Replace <file path> with the path to the video source

### 2. For video flipping

```
gst-launch-1.0 filesrc location=<file path> ! decodebin ! videorga
flip = <none/90R/90L/180> ! ximagesink
```

### 3. For video scaling

```
gst-launch-1.0 filesrc location=<file path> ! decodebin ! videorga !
video/x-raw, width=x, height=y ! ximagesink
```

### 4. For video scaling with video flip

```
gst-launch-1.0 filesrc location=<file path> ! decodebin ! videorga
flip = <none, 90R, 90L, 180 > ! video/x-raw, width=x, height=y !
ximagesink
```

### 5. For encoding

```
gst-launch-1.0 videotestsrc num-buffers = 100 ! video/x-raw,
width=1920, height=1080 ! filesink location=<file
path>out_1080p.h264Replace <file path> with the path to store the encoded
stream
```

## 5.1 Enabling Audio on SoFIA

This section describes the steps to enable audio decoders in the system image.

Set the commercial flag in the configuration file and also include the `gstlibav` file in the system image by following these steps:

### 1. Update the `local.conf` file as follows:

```
$ vi /<path to sofia-oe-buildsystem/build-sofia-
3gr>/conf/local.conf "
```

```
LICENSE_FLAGS_WHITELIST += "intel-commercial"
```

```
LICENSE_FLAGS_WHITELIST += "commercial"
```

Add/update "LICENSE\_FLAGS\_WHITELIST" flag to commercial as shown above.



2. Build Gstreamer by executing “bitbake gstreamer1.0-libav”.
3. Copy the updated libgstlibav.so file to the target system.

This library is generated at following path in the build system.

```
" /<path to>/sofia-oe-buildsystem/build-sofia-3gr/tmp-jethro-sofia-3gr/sysroots/sofia-3gr/usr/lib/gstreamer-1.0 "
```

4. Copy the generated library file (libgstlibav.so) to the following location of the target system.

```
/usr/lib/gstreamer-1.0/
```

5. Reboot.
6. Execute the following commands to enable the onboard audio:

```
$ amixer -D hw:0 cset name='EP Enable Switch' 1 > /dev/null
$ amixer -D hw:0 cset name='LS Output Route' 1 > /dev/null
$ amixer -D hw:0 cset name='PCM Audio Playback' 1 > /dev/null
$ amixer -D hw:0 cset name='PCM Audio Record' 1 > /dev/null
$ amixer -D hw:0 cset name='LS Output Route', 1 > /dev/null
$ amixer -D hw:0 cset name='DMIC1 Enable Switch', 1 > /dev/null
$ amixer -D hw:0 cset name='DMIC2 Enable Switch', 1 > /dev/null
```

7. Execute the following commands to route the audio to the 3.5-mm audio jack.

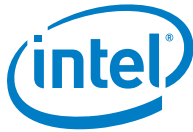
```
amixer -c 0 cset name='HSL Enable Route' 1 > /dev/null
amixer -c 0 cset name='HSR Enable Switch' 1 > /dev/null
amixer -c 0 cset name='HSL Output Route' 1 > /dev/null
amixer -c 0 cset name='Headset Gain' 16 > /dev/null
```

**Note:** The following sample is for video in MPEG 4 format and audio in AAC format.

```
gst-launch-1.0 filesrc location=<file_path> ! decodebin name=dec dec.
! audioconvert ! alsasink dec. ! videorga flip=<rotation> !
ximagesink

gst-launch-1.0 filesrc location=<file_path> ! decodebin name=dec dec.
! audioconvert ! alsasink dec. ! videorga flip=<rotation> !
waylandsink
```

**Note:** Replace <file path> with the path to the video source.



## Appendix A

---

### A.1 Workaround for Issue 1804305651

Follow these steps to put the system into deep sleep:

1. To set the modem into flight mode, send a command from the serial UART terminal.

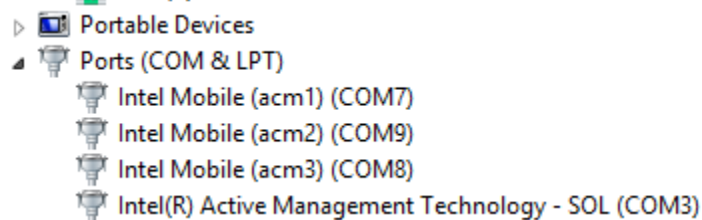
```
$ python /usr/lib/ofono/tests/offline-modem
```



```
Yocto Project Meta Distro for Intel(R) SoFIA 2.0+snapshot-20160919 sofia-3gr tty
S0

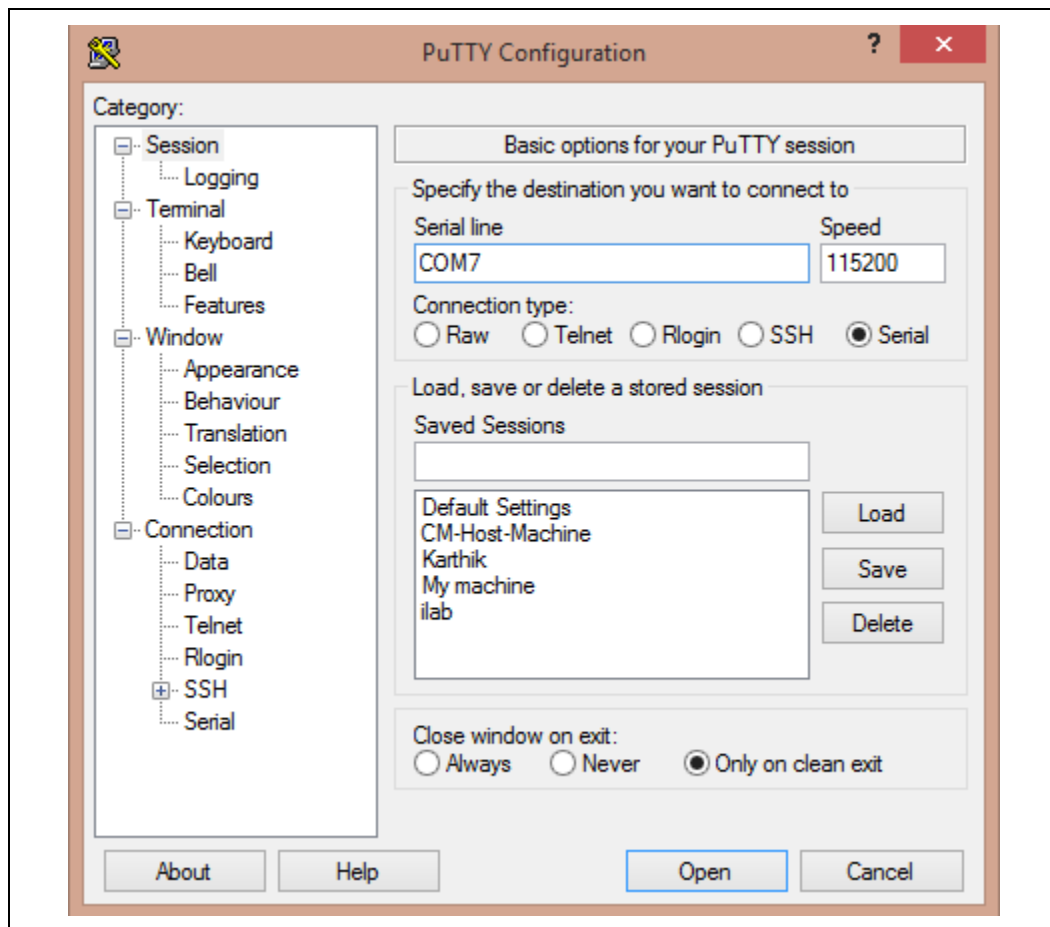
sofia-3gr login: root
root@sofia-3gr:~# python /usr/lib/ofono/test/offline-modem
Setting modem /ril_0 offline...
root@sofia-3gr:~#
```

- a. The “AT” command has to be sent from the ACM1 COM PORT. Connect the board to the laptop USB port and the enumerated COM PORT appears as follows:



- b. Choose ACM1 to send the AT Command (unplug the USB cable).



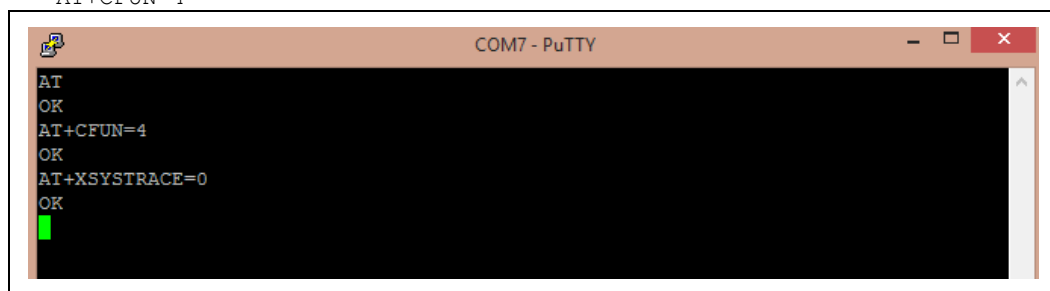


2. To disable the modem tracing:

```
AT+XSYSTRACE=0
```

3. To turn off the transceiver:

```
AT+CFUN=4
```



4. To stop the AFE:

```
$ systemctl stop sofia-afe
```

**Note:** Kill sofia-afe if it is still running.



5. To disable the magnetometer's power:  

```
$ echo disable > /sys/kernel/debug/devpm/ext_magnetometer
```
6. To disable the power to the PWM:  

```
$ echo disable > /sys/kernel/debug/devpm/pwm
```
7. Suspend the Yocto Project\* system:  

```
$ echo mem > /sys/power/state
```
8. Observe the power at the power profiler.
9. Wake up the system by pressing the PWR button next to the rest button.
10. To check the system's deep sleep count:  

```
$ cat /sys/kernel/debug/mid_pmu_states
```

Below shows the example:

```
root@sofia-3gr:~# cat /sys/kernel/debug/mid_pmu_states
```

[ 608.971462]	time(secs)	residency(%)	count	Avg.
Res (Sec)				
s3		16.442		2.631
1	0.020			
Total time: 608.985 Sec				

## A.2 Workaround for Issue 1504330812

### //AT CFUN functionality

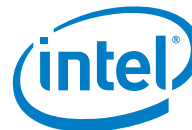
1. AT+CFUN=0 -> Modem Shutdown
2. AT+CFUN=1 -> Modem Radio on full functionality
3. AT+CFUN=4 -> Modem Radio off min functionality
4. AT+CFUN=33 -> CPS Stack off
5. AT+CFUN=34 -> CPS stack on

### //Ofono Functionality

1. online-modem -> CPS stack on
2. offline-modem -> CPS stack off
3. enable-modem -> Radio on (Airplane mode)
4. disable-modem -> Radio off

### //Interworking of AT and ofono

1. It is not advised to mix AT commands with ofono-commands. AT commands are to be used only when ofono does not provide same functionality.
2. +CFUN: 4 should be treated as Airplane Mode.



No functional mode changes are allowed in this mode except AT+CFUN = 0/1.

3. **offline-modem and online-modem result in error if +CFUN: 4.**

Ensure +CFUN: 1 for offline-modem and online-modem success.

4. **disable-modem provides the same functionality as +CFUN: 4.**

This should be preferred over AT command for ofono testing.

5. **After disable-modem, issue enable-modem before giving offline-modem and online-modem request.**

## §