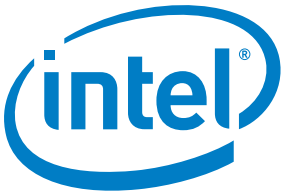

Intel Atom® Processor E3900 SoC Family/ Intel® Celeron® Processor N3350/ Intel® Pentium® Processor N4200/ Intel® Celeron® Processor J3355 & J3455 for Yocto Project*

Getting Started Guide

October 2018

Document Number: 334828-002US

Contents

Revision History

Date	Revision	Description
October 2018	002	Updated product marketing name. Added reference documents for MR4 Release. Added known issues.
February 2017	001	Initial release.

§

1.0 Introduction

This Guide contains general information and guidelines to prepare the build environments in the build system. This Getting Started Guide targets users who need to build, customize, and configure their build environment specifically for Yocto* Board Support Systems (BSPs).

1.1 Terminology

Table 1. Terminology

Term	Description
BSP	Board Support Package
OS	Operating System
RAM	Random Access Memory
SoC	System-on-a-Chip

1.2 Reference Documents

Table 2. Reference Documents

Document	Document No./Location
Intel Atom® Processor E3900 SoC Family/Intel® Celeron® Processor N3350/Intel® Pentium® Processor N4200/ Intel® Celeron® Processor J3355 & J3455 Board Support Package for Yocto Project - MR3.1 – Release Notes*	333732
Intel Atom® Processor E3900 SoC Family/Intel® Celeron® Processor N3350/Intel® Pentium® Processor N4200/ Intel® Celeron® Processor J3355 & J3455 Board Support Package for Yocto Project - MR4 – Release Notes*	595926

1.3 Known Issues

Table 3. Known Issues

Reference No.	Issue	Workaround
1504693541	Desktop manager crashes and restarts during image build.	If Yocto Project* 2.3 or later is used, access build system and build image using SSH instead of directly using the GUI on the build system. This issue is tracked on Bugzilla*.

2.0 Setting up the Build System

This chapter contains the list of pre-requisites and the steps to configure the build environment to build the Yocto Project*-based image for Apollo Lake. Once the build system is properly set up and configured, it can be used to build the image that will eventually be used to boot up the Apollo Lake-based platform. Refer to the *BSP Release Notes* on how to get the Apollo Lake BSP and build the image.

2.1 Pre-requisites

Prepare a build system (computer) with the recommended minimum hardware requirement:

- Intel® Core™ i7 processor (4 cores with Intel® Hyper-Threading Technology)
- Minimum of 4 GB Random Access Memory (RAM)
- Minimum of 250 GB disk space
- High-speed network connectivity
- Linux* OS for Yocto Project* BSP: Ubuntu* v14.04 LTS

2.2 Preparing the Build Environment in Build System

The following procedure is used to prepare the build environment in the build system.

1. Set user as **sudoer**, so the user can perform super-user tasks:
   a. Change to supervisor:
      ```
      $ sudo su
      ```
   b. modify /etc/sudoers by adding the highlighted value:
      ```
      $ gedit /etc/sudoers
      root ALL=(ALL:ALL) ALL

      <username> ALL=(ALL:ALL) ALL
      ```
      Note: *<username> is the name that is used to log in to the build system.*
   c. Use the following to change back to normal user:
      ```
      $ su - <username>
      ```

Note: Step 1 is usually performed by the network administrator.

2. Perform this step only if the build system is behind a corporate network with a proxy server.

   a. Add the following lines in the `/etc/environment`:

   ```
   $ sudo gedit /etc/environment
   export SOCKS_SERVER=socks://<proxy server IP or DNS>:<socks port number>
   export HTTP_PROXY=http://<proxy server IP or DNS>:<http port number>
   export HTTPS_PROXY=https://<proxy server IP or DNS>:<https port number>
   export FTP_PROXY=http://<proxy server IP or DNS>:<ftp port number>
   ```

   b. Edit the .bashrc file and append to the proxy settings using the following:

   ```
   $gedit ~/.bashrc

   # Insert the below lines in ~/.bashrc

   export SOCKS_SERVER=socks://<proxy server IP or DNS>:<socks port number>

   export HTTP_PROXY=http://<proxy server IP or DNS>:<http port number>

   export HTTPS_PROXY=https://<proxy server IP or DNS>:<https port number>

   export FTP_PROXY=http://<proxy server IP or DNS>:<ftp port number>

   export export HTTP_DIRECT=localhost,127.0.0.0/8,<your network subnets>

   export SOCKS_DIRECT=$HTTP_DIRECT

   export NO_PROXY=$HTTP_DIRECT

   export ALL_PROXY=$HTTP_DIRECT

   export socks_server=$SOCKS_SERVER

   export http_proxy=$HTTP_PROXY

   export https_proxy=$HTTPS_PROXY

   export ftp_proxy=$FTP_PROXY

   export http_direct=$HTTP_DIRECT

   export socks_direct=$SOCKS_DIRECT

   export no_proxy=$NO_PROXY
   ```

c. Reload the bash environment using the following:

```
$source ~/.bashrc
```

d. Set up the proxy for **apt-get** by editing **apt.conf** using the following:

```
$ sudo gedit /etc/apt/apt.conf
Acquire::http::proxy "http://<proxy server IP or DNS>:<http
port>/";
Acquire::https::proxy "https://<proxy server IP or
DNS>:<https port>/";
Acquire::ftp::proxy "ftp://<proxy server IP or DNS>:<ftp
port>/";
Acquire::socks::proxy "socks://<proxy server IP or
DNS>:<socks proxy>/";
```

3. Use the following commands to install the necessary tools:

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

4. Create and add the following configuration `under /home/<username>/.gitconfig` by using the following:

```
[user]
email = <your.name>@<your_domain>.com
name = <Your Name>
 [sendemail]
smtpserver = <Your organization's SMTP server address>
signedoffcc = false
suppresscc = all
chainreplyto = false
assume8bitEncoding = utf-8
from = <Your Name> <your.name@your_domain.com>
confirm = always
 [color "grep"]
match = red
 [color]
diff = auto
ui = auto
interactive = auto
grep = always
 [alias]
co = checkout
br = branch
ci = commit
st = status
ol = log -oneline
 [core]
editor = gedit OR vi
#uncomment the gitproxy variable below if you require a
gitproxy
#gitproxy = /home/<username>/bin/gitproxy
```

5. If the build machine is behind a corporate network with a proxy server, create the `/home/<username>/bin/gitproxy` file and insert the SOCKS proxy configuration using the following:

```
$ gedit /home/<username>/bin/gitproxy
#!/bin/bash
exec socat stdio SOCKS: <proxy server IP or DNS>:$1:$2
#Make gitproxy executable.
$ chmod +x /home/<username>/bin/gitproxy
```

6. Generate an SSH key and add the key to your GitHub* account. Go to this link and follow the instructions on each subsection on the webpage.
   i. Checking for existing SSH Keys
   ii. Generating a new SSH Key and adding it to the ssh-agent
   iii. Adding a new SSH key to your GitHub account
   iv. Testing your SSH connection
7. Create and add the following lines in `~/.ssh/config` by using the following:

```
host github.com
user git
hostname ssh.github.com
identityfile ~/.ssh/id_rsa
port 443
```

8. If the build machine is behind a corporate environment, add the following line to ~/.ssh/config to the following:

```
proxycommand /bin/nc -X connect -x <proxy server IP or
DNS>:<http port number> %h %p
tcpkeepalive yes
compression yes
connectionattempts 3
```

9. Reboot the build system.
10. You are now ready to clone the Intel Atom® Processor E3900 SoC Family/Intel® Celeron® Processor N3350/Intel® Pentium® Processor N4200/ Intel® Celeron® Processor J3355 & J3455 Yocto Project*-based BSP from the GitHub repository and build the BSP image. Refer to Table 2, *Intel Atom® Processor E3900 SoC Family/Intel® Celeron® Processor N3350/Intel® Pentium® Processor N4200/ Intel® Celeron® Processor J3355 & J3455 Yocto Project* BSP* for more details.

§