Linux BSP 36.0_cd Quick Start Guide for S32G3

07-Feb-2023

Contents

1 Introduction

This document explains how to build and install the NXP Linux Board Support Package (BSP) on S32G3 platform and boards. All steps needed to get the S32G3 platform and boards running are detailed, including board DIP switch settings, steps to download an OS image, and instructions on configuring and using the U-Boot bootloader and ARM Trusted Firmware (TF-A) implementation for S32G3 .

2 Linux BSP installation

2.1 Getting the Yocto image

All the steps described below have been run and validated on Ubuntu-20.04 LTS (native or through a virtual machine). It is then recommended to install Ubuntu-20.04 LTS before going through the following sections.

To get the BSP you need to have **repo** installed and its prerequisites. This only needs to be done once.

Update the package manager:

```
sudo <pkg-mgr> update
```

Install dependencies:

- python 2.x - 2.6 or newer:

  ```
  sudo <pkg-mgr> install python
  ```

- git 1.8.3 or newer:

  ```
  sudo <pkg-mgr> install git
  ```

- curl:

  ```
  sudo <pkg-mgr> install curl
  ```

  where <pkg-mgr> is the package manager for your distribution (apt-get or apt for Debian/Ubuntu, yum or dnf for CentOS/Fedora, zypper for SUSE).

Install the **repo** utility:

```
mkdir ~/bin
curl http://commondatastorage.googleapis.com/git-repo-downloads/repo > ~/bin/repo
chmod a+x ~/bin/repo
PATH=${PATH}:~/bin
```

Configure your git environment (you may skip this option if you have git already configured):

```
git config --global user.email "john.doe@example.com"
git config --global user.name "John Doe"
```

Next, download the Yocto Project Environment into your directory:

```
mkdir fsl-auto-yocto-bsp
cd fsl-auto-yocto-bsp
repo init -u https://github.com/nxp-auto-linux/auto_yocto_bsp -b release/bsp36.0_cd
repo sync
```

This will download the sources for the latest NXP Auto Linux BSP (from the branch release/bsp36.0_cd), structured on top of the Yocto Kirkstone release and upstream NXP QorIQ SDK.

2.2 Build Linux BSP

Enter the directory *fsl-auto-yocto-bsp* and follow below instructions (available in the *README* file).
Note:
A Yocto build needs at least 50GB of free space and takes a lot of time (2 to 10 hours, depending on the system configuration). It is recommended to use a powerful system with many cores and a fast storage media (SSD is recommended). The recommended RAM size is 8 GB.

This release includes support for:

- machines: s32g399aevb3, s32g398aevb3, s32g379aevb3, s32g378aevb3, s32g399ardb3
- images: fsl-image-base, fsl-image-auto

To build the Linux BSP, please follow the steps:

1. First time setup

   ```
   sudo apt update
   ```

   ```
   ./sources/meta-alb/scripts/host-prepare.sh
   ```

2. Creating Build Directories

   Now you can create a build directory in the SDK root with:

   ```
   source nxp-setup-alb.sh -m <machine>
   ```

   where <machine> can be any of the supported machines: s32g399aevb3, s32g398aevb3, s32g379aevb3, s32g378aevb3, s32g399ardb3. For example, if targeting the S32G3 EVB board, machine is s32g399aevb3:

   ```
   source nxp-setup-alb.sh -m s32g399aevb3
   ```

3. Adding optional features

   It is possible to add optional features to the Linux BSP. The required steps are documented into the User Manual.

4. Build the image

   When the above is done, a `bitbake <imagename>`, e.g.,

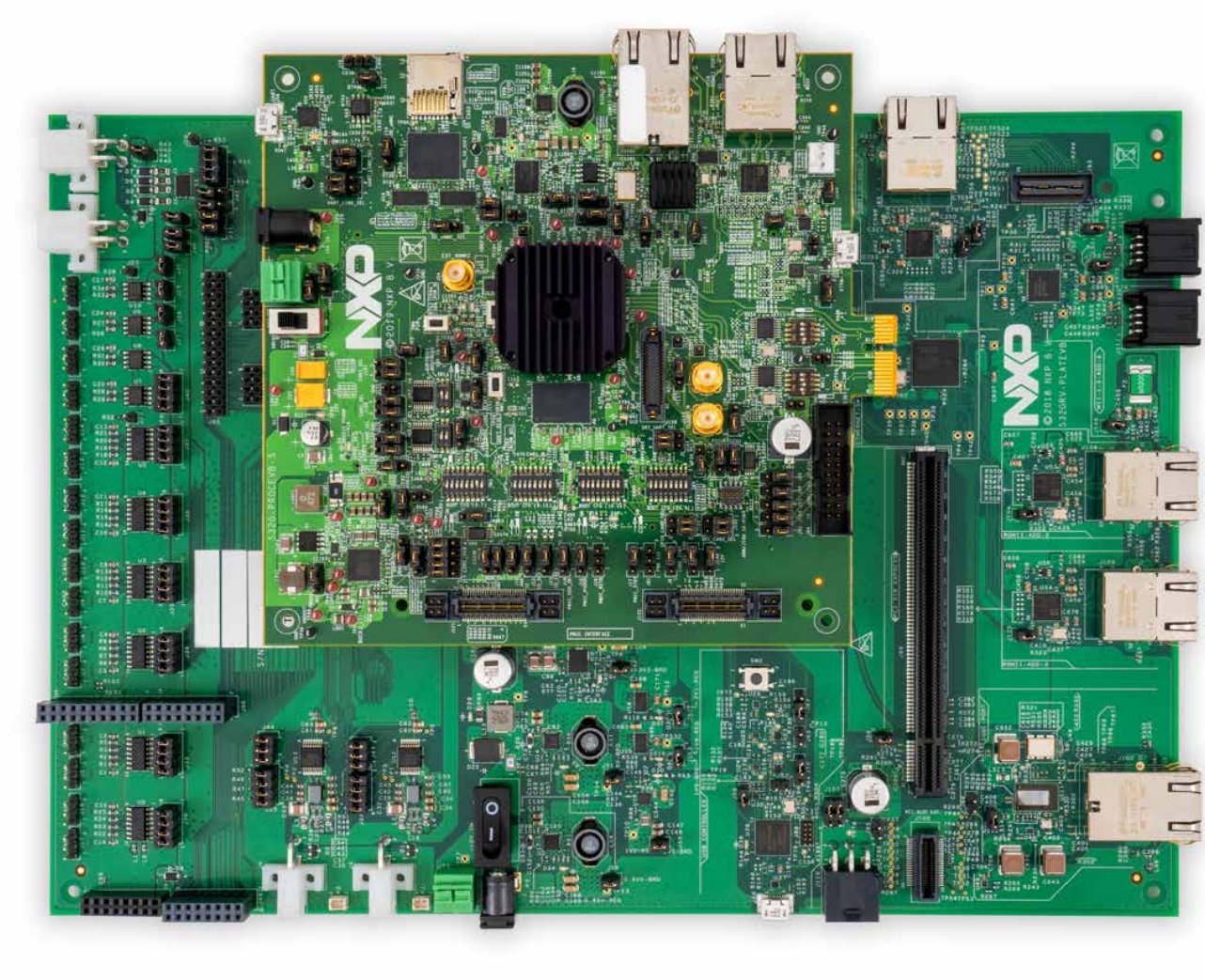
   ```
   bitbake fsl-image-auto
   ```

   would be enough to completely build ATF, U-Boot, Linux Kernel, modules and a root filesystem ready to be deployed.

After a successful build, the results will be placed in the *build_s32g399aevb3/tmp/deploy/images/s32g399aevb3* directory, if the chosen machine was s32g399aevb3. Results for other machines will be placed in similar directories, but named according to the machine name.

3 Board setup

3.1 S32G3 EVB

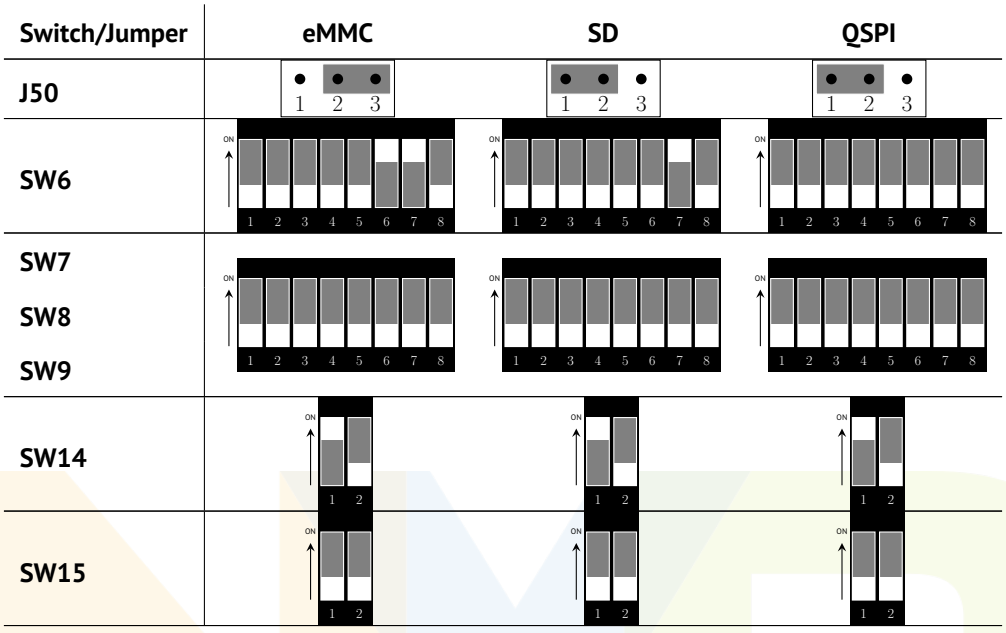


3.1.1 Jumper settings

The current pinmuxing configuration for UART using LF0 requires that jumpers J124 on the S32G-PROCEVB-S board be in the 1-3, 2-4 position, which is their default position but is different from BSP22.5.

Jumpers J159 should be in the 1-3, 2-4 position (default configuration).

Below are the details for RCON booting from SD/eMMC/QSPI. This assumes the FUSE_SEL fuse on the chip has not been blown. For more details regarding the boot modes supported, please refer to the SoC Reference Manual



3.1.2 Setup the SD Card

The .sdcard format creates an image with all necessary partitions and loads the bootloader, kernel and rootfs to this image.

Ensure that any partitions on the card are properly unmounted before writing the card image, or you may have a corrupted card image in the end. Also ensure to properly "sync" the filesystem before ejecting the card to ensure all data has been written.

In this example, the device node assigned is DEVSD (a block is 1kB large).

If DEVSD is /dev/sdc, we should execute the following command:

```
export DEVSD=/dev/sdc
```

You can just low level copy the data on this file to the SD Card device using dd as on the following command example:

```
sudo dd if=<image name>.sdcard of=${DEVSD} bs=1M && sync
```

Note:

Yocto supports the generation of both uImage and Image kernel binary formats, but only the 'Image' format works with the current configuration of U-Boot. It is not expected that uImage support will be restored in the future.

3.1.3 Booting

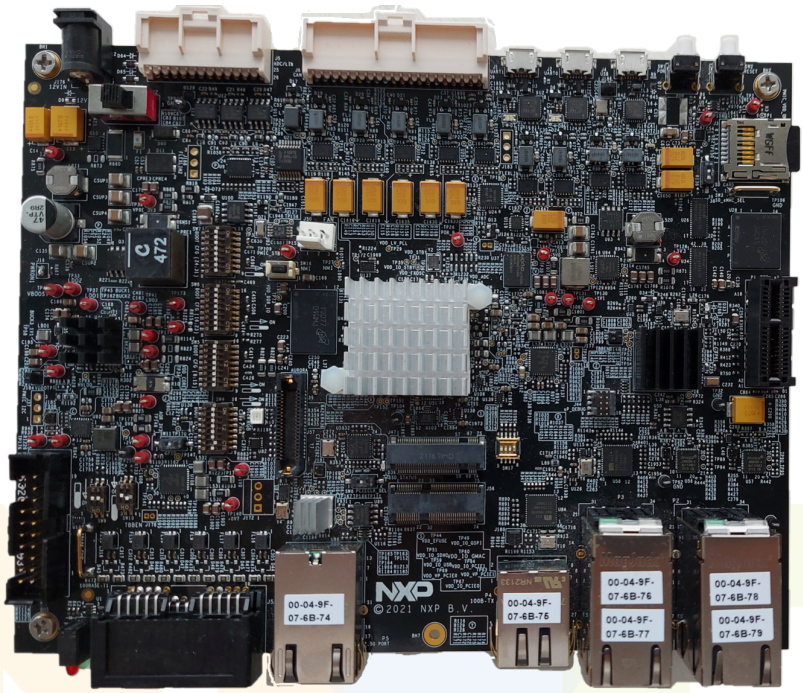
1. Insert a bootable SD Card into the microSD Card slot (J34) on the top of the board.
2. Configure the boot switches to boot from SD.
3. Connect the micro USB cable from a free usb port on your host PC to the J58 connector labeled “UART1”.
4. Start your favorite terminal software (such as Minicom, TeraTerm or Putty) on your host PC and configure it for 115200 baud, 8 data bits, no parity, and 1 stop bit with no handshake.
5. Plug power supply (P1) and turn on the power switch (SW10)
6. On powering on, you will see the console output on your terminal window. If everything was done correctly the board should boot completely into Linux. The default login account is root with an empty password.

3.2 S32G3 EVB3

In case of S32G3 EVB3 SCH-50784 REV A, the same steps to configure the boot jumpers and switch configuration apply as on the board S32G3 EVB SCH-32170 REV B1. Same applies to power source selection, current pinmuxing configuration for UART using LF0 and booting from parallel RCON.

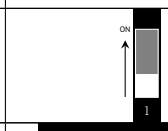
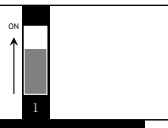
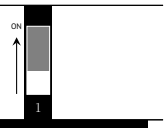
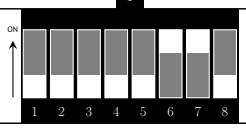
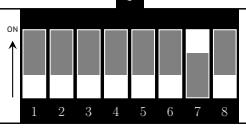
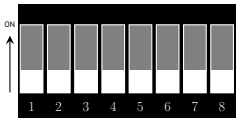
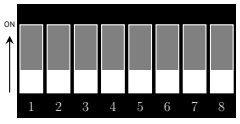
In addition, S32G3 EVB3 SCH-32170 REV B1 provides UART1 serial interface.

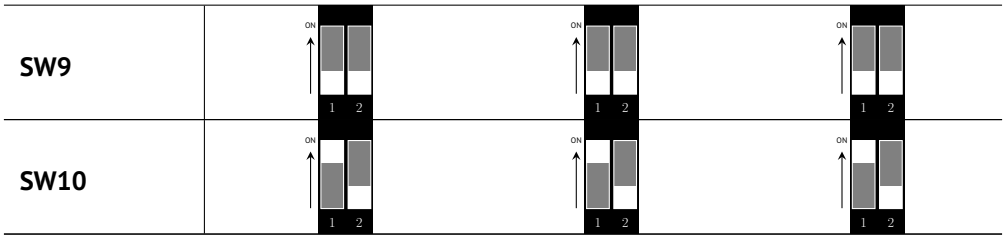
3.3 S32G399A RDB3



3.3.1 Jumper settings

This chapter describes the configuration of the switch for the S32G399A RDB3 SCH-53060 REV E board.

Switch/Jumper	eMMC	SD	QSPI
SW3			
SW4			
SW5 SW6 SW7			



3.3.2 Setup the SD Card

For the S32G399A RDB3 board, the same instructions apply as for the S32G3 EVB board. See details in section 3.1.2.

3.3.3 Booting

1. Insert a bootable SD Card into the microSD Card slot on the side of the board.
2. Configure the boot switches to boot from SD.
3. Connect the micro USB cable from a free usb port on your host PC to the UART0 connector.
4. Start your favorite terminal software (such as Minicom or TeraTerm) on your host PC and configure it for 115200 baud, 8 data bits, no parity, and 1 stop bit with no handshake.
5. Plug power supply and turn on the power switch.
6. On powering on, you will see console output on your terminal window. If everything was done correctly the board should boot completely into Linux. The default login account is root with an empty password.