How-to: Implement and Test SPIDEV on the phyCORE-AM57x Linux RDK

phyCORE-AM57x Rapid Development Kit

The following will provide instructions for exercising SPI1 on the phyCORE-AM57x RDK using release PD17.1.0. This example is a loop-back test and will be implemented with the phyCORE-AM57x RDK and expansion board. Loop-back will be done at the expansion board header by connecting SPI1_DIN to SPI1_DOUT. Implementation details are given simply for informative purposes and are implemented in PD17.1.0.

Using a pre-built image

Loading a pre-built image to SD card

1. The SPIDEV functionality described in this how-to is implemented in PD17.1.0. Download the pre-built image binaries from here.
2. Follow the instructions in the Quickstart guide to partition and flash the SD card with the pre-built image.

Building spidev_test.c

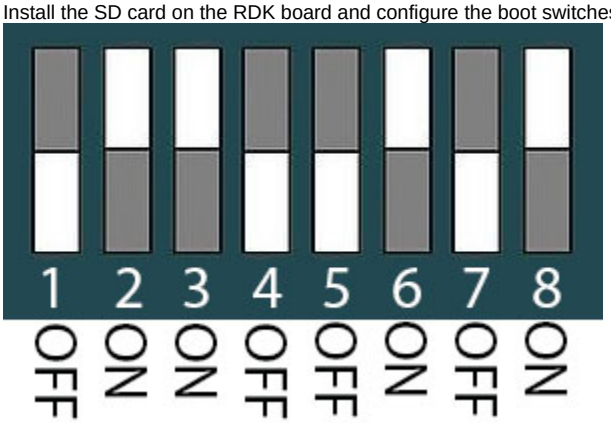
1. spidev_test.c is an example included in the Linux kernel documentation. The file can be downloaded here.
2. Execute the following commands on the development host to cross-compile the test program so that it may be run on your custom BSP. The resulting file is called "spidev_test" and is located in the directory specified by the "-o" option.

```
#Get and extract the Linaro Toolchain
cd ~/Documents
wget https://releases.linaro.org/components/toolchain/binaries/5.3-2016.02/arm-linux-gnueabihf/gcc-linaro-5.3-2016.02-x86_64_arm-linux-gnueabihf.tar.xz
tar -Jxvf gcc-linaro-5.3-2016.02-x86_64_arm-linux-gnueabihf.tar.xz -C ./

#Add the toolchain to PATH
export PATH=./gcc-linaro-5.3-2016.02-x86_64_arm-linux-gnueabihf/bin:$PATH

#Build spidev_test.c
arm-linux-gnueabihf-gcc -O3 -o spidev_test spidev_test.c
```

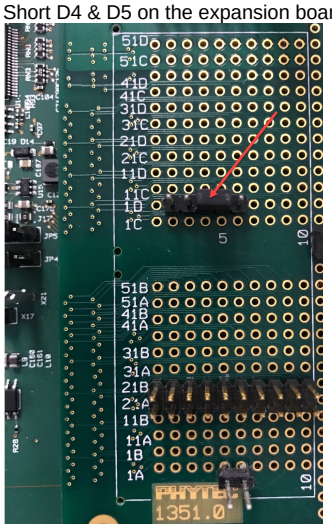
3. Mount the SD card that contains the BSP image and copy the compiled spidev_test to the root directory.

Hardware setup

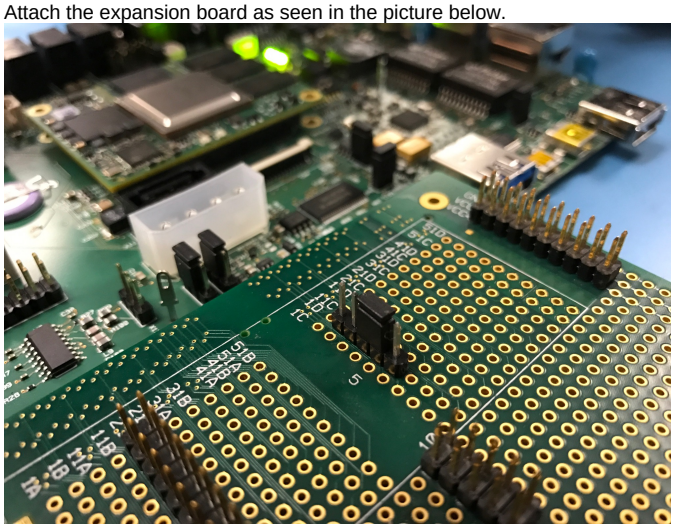
1. Install the SD card on the RDK board and configure the boot switches (S5) to boot from SD card.



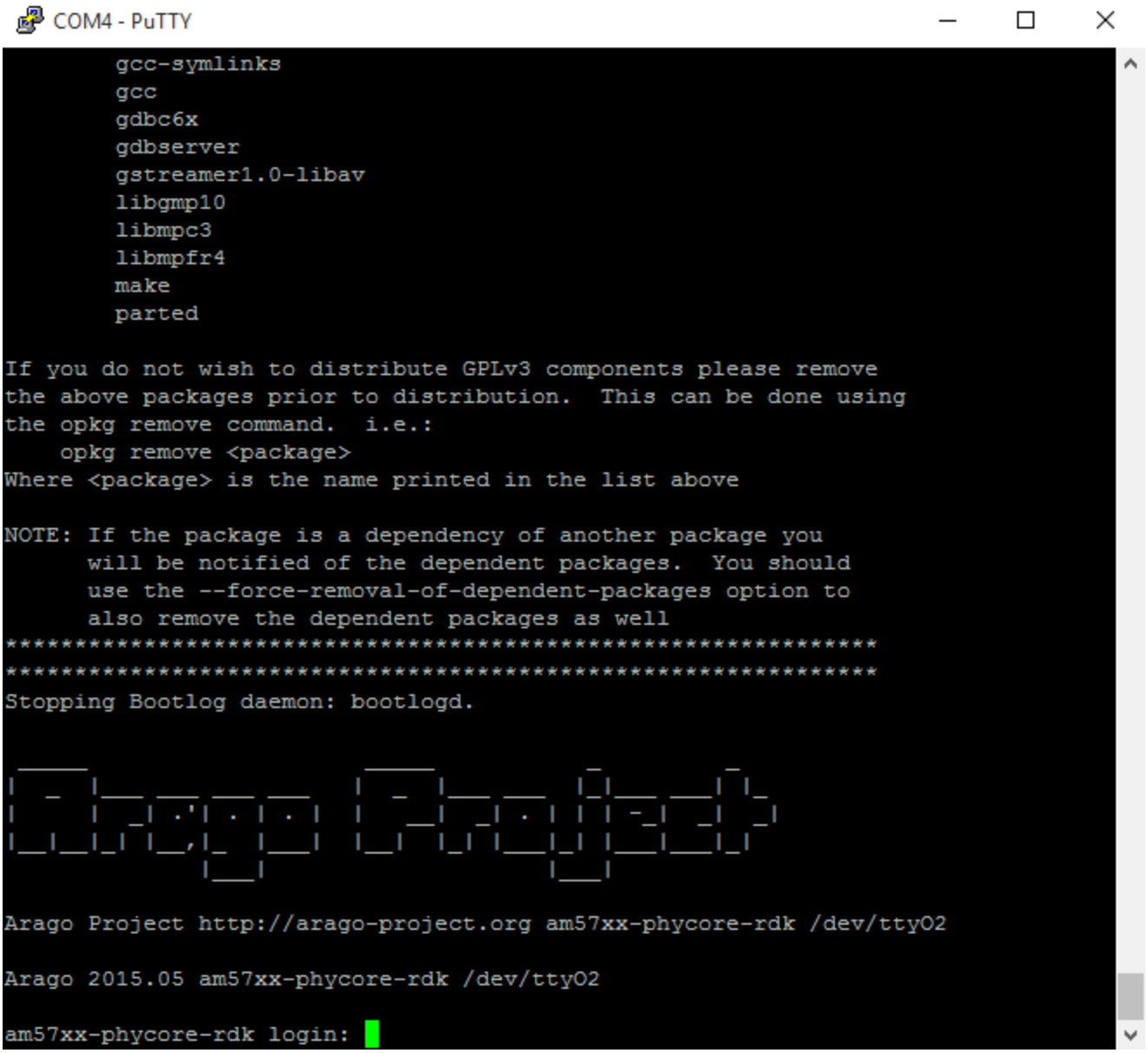
2. Short D4 & D5 on the expansion board as seen in the picture below.



3. Attach the expansion board as seen in the picture below.



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 (8n1) with no handshake.
5. Apply power and press the power switch (S2) to boot Linux
6. At the login prompt, enter "root" as the username. There is no required password.



Executing the application

Execute the following commands from Linux, through the UART debug terminal, to run the test application:

```
cd /
./spidev_test -v -D /dev/spidev1.0
```

A successful loop-back test will result with the following output. Note that the "TX" and "RX" lines match, indicating that the loop-back was successful.

```
root@am57xx-phycore-rdk:/# ./spidev_test -v -D /dev/spidev1.0
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF F0 0D  | ......@.....................
RX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF F0 0D  | ......@.....................
```

Implementation details

The changes described below have been included in PD17.1.0. However, if you are using an older BSP release (ex. PD16.1.0) or want to add another SPI interface the below steps can be used as a reference.

Configuring the device tree

To enable the SPI1 interface, modifications to the device tree will be required. The following instructions show how to modify the files specific for the phyCORE-AM57x Rapid Development kit to multiplex the signals and enable the interface. The device tree files that will need to be modified are located in this directory:

```
/opt/PHYTEC_BSPs/yocto_ti/build/arago-tmp-external-linaro-toolchain/work/am57xx_phycore_rdk-linux-gnueabi/linux-phytec-ti/4.4.12+git_v4.4.12-phy1-r7a/git/arch/arm/boot/dts
```


Note: The path to the Linux source above may differ depending on the release used.

1. Multiplex the signals for the SPI1 interface by adding the following to "dra7_pmx_core" in am57xx-phycore-som.dtsi:

```
mcspi1_pins_default: mcspi1_pins_default {
        pinctrl-single,pins = <
    0x3a4 (PIN_INPUT | MUX_MODE0)         /* spi1_sclk */
    0x3a8 (PIN_INPUT | MUX_MODE0)         /* spi1_d1   */
    0x3ac (PIN_INPUT | MUX_MODE0)         /* spi1_d0   */
    0x3b0 (PIN_OUTPUT_PULLUP | MUX_MODE0) /* spi1_cs0  */
    0x3b4 (PIN_OUTPUT_PULLUP | MUX_MODE0) /* spi1_cs1  */
        >;
};
```

2. Configure the pins for the SPI1 interface by adding the following to am57xx-phycore-som.dtsi:

```
&mcspi1 {
        status = "disabled";
        pinctrl-names = "default";
        pinctrl-0 = <&mcspi1_pins_default>;

        ti,pindir-d0-out-d1-in;
};
```

3. Configure the SPI interface by adding the following to am57xx-pcm-948.dtsi:

```
&mcspi1 {
        spidev1_0: spidev1@0 {
            compatible = "linux,spidev";

        reg = <0>;
        spi-max-frequency = <48000000>;
    };
};
```

4. Enable the SPI device by adding the following to am57xx-phycore-rdk.dts:

```
&mcspi1 {
        status = "okay";
};
```

Configuring the kernel

Now that we have configured the SPI interface and pins, we need to enable the driver in the kernel so we can access the interface from user space. The following instructions show how to configure the Linux kernel used on the phyCORE-AM57x Rapid Development kit to enable spidev access in user space.

1. Verify your $YOCTO_DIR environmental variable is still set, if not execute the following command:

```
cd /opt/PHYTEC_BSPs/yocto_ti
export YOCTO_DIR=`pwd`
```

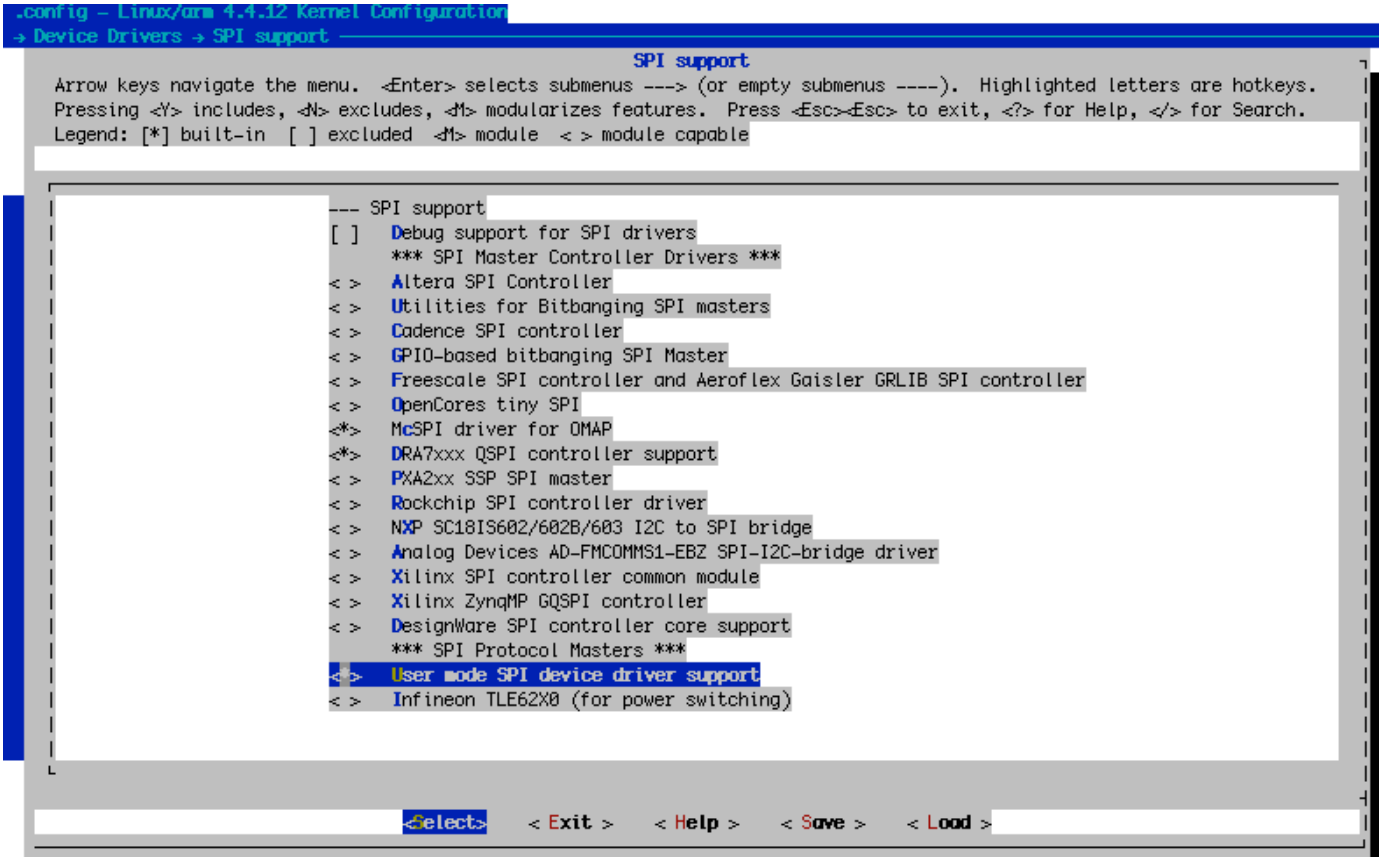
2. Verify the toolcahin is in PATH, if not, execute the following command:

```
export PATH=/opt/PHYTEC_BSPs/gcc-linaro-5.3-2016.02-x86_64_arm-linux-gnueabihf/bin:$PATH
```

3. Add the SPI user space driver using menuconfig

```
cd $YOCTO_DIR/build
bitbake linux-phytec-ti -c menuconfig
```

4. Enable built-in support for "User mode SPI device driver support" as shown in the picture below. This can be found in the Device DriversSPI Support



Rebuilding the Device Tree and Kernel

After the necessary modifications to the kernel configuration and device tree, we need to rebuild the kernel and root filesystem.

1. Rebuild the kernel:

```
cd $YOCTO_DIR/build
bitbake linux-phytec-ti -f -c compile && bitbake linux-phytec-ti
```

2. Rebuild the root filesystem

```
cd $YOCTO_DIR/build
bitbake arago-core-tisdk-image
```