GRAND CHALLENGE - 2025

SHAKTI SDK USER MANUAL

DEVELOPED BY :
SHAKTI DEVELOPMENT TEAM @ IITM
SHAKTI.ORG.IN

0.1 Proprietary Notice

The project was funded by Ministry of Electronics and Information Technology (MeitY), Government of India

0.2 Release Information

Version	Date	Updates
1.0	24/04/2025	Initial version

Table of Contents

1. SHAKTI-SDK

Software Development Kits (SDKs) are an integral part in the development of any configurable device. The main objective behind using a SDK is to reduce the development time. The SHAKTI-SDK is a platform that enables developing applications over the SHAKTI class of processors. We provide firmware support for end users to develop applications. The SHAKTI-SDK is simple and easily customizable. Some of the essential features like DEBUG codes and board support libraries are provided.

2. Board Details

We provide support for development of Shakti based SoC on Arty7 100T FPGA boards. Below section gives detailed information on the board.

2.1 Yamuna

– SoC based on 32 bit, 6 stage SHAKTI C class.
– Supported on the Artix 7 100T board.
– Compliance with RV32IMAC.

Arty A7 - User manual
https://reference.digilentinc.com/reference/programmable-logic/arty-a7/Reference-manual

3. Board setup

3.1 Powering the board

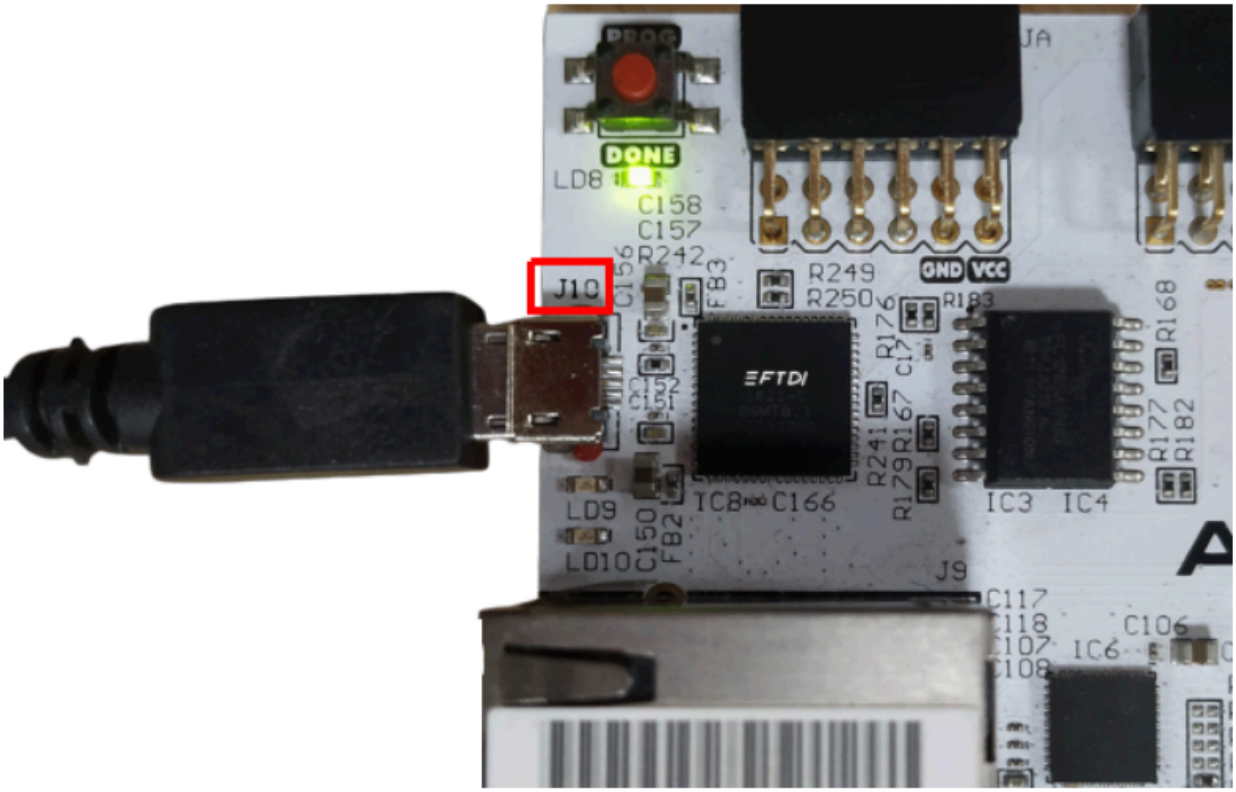
Plug one end of the micro USB cable into the PC's slot and the other end to the MicroUSB connector (J10) in the board. This will power ON the board. please see Figure 1. The connector J10 is a JTAG and UART port combination. If a sensor requires more power, an external 12V power supply can be connected via Power Jack (J12) . Refer Arty reference manual for detailed power on instructions.

3.2 Setting up the Debugger

This section explains setting up the board for debug mode. The setup for standalone mode is discussed in the Section 5.5 of this manual. The debugger for the board is the Xilinx FTDI chip on the Arty boards. The details on how to connect the debugger to the board is given below.

3.3 Debug interface over Xilinx FTDI (recommended)

The FPGA board is powered on by connecting the micro USB to pin J10. This also connects internally to the UART0 via FTDI interface which provides debugger support.



4. Software Development Flow

4.1 SHAKTI-SDK Architecture

The SHAKTI-SDK is a C/C++ platform to develop applications over SHAKTI. The SDK has the necessary firmware code and framework to develop newer applications on the hardware. The framework is lightweight and customizable.

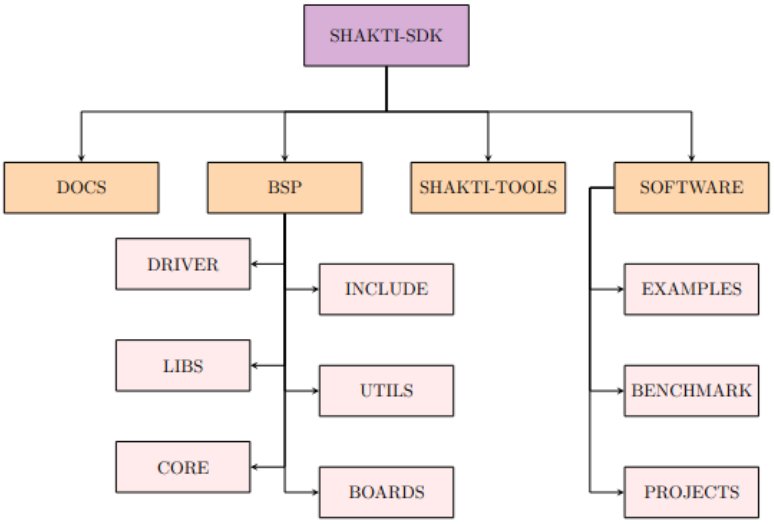


Figure 5: SDK architecture

4.2 Board Support Package

The BSP consists of driver files for various devices and system files. It contains certain platform dependent definitions for each board. Essentially, the BSP is the layer above the hardware. It includes the following sub directories,

4.2.1 Drivers

The drivers are a set of software constructs that help software applications to access the devices in the SoC. They are generally low level API, that execute a particular task in the hardware.

4.2.2 Include

This folder has header files for core and each driver. The board independent variable/macro definitions and declarations pertaining to each driver is included here.

4.2.3 Libs

The library utilities, boot code are hosted here. Library is a common place for reusable code. The libraries can be compiled as a separate "lib" file and used.

4.2.4 Core

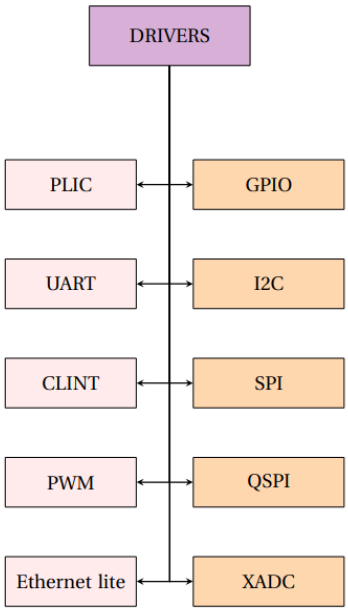
The core usually has functions related to the startup codes, Trap handlers and interrupt vectors.The codes related to memory initialisation are also available here.

4.2.5 Utils

This contains the code related to the standalone mode feature of the shakti processor.

4.2.6 Third_party

This folder provides support for external boards as well as custom boards. Includes the definitions of board specific functions such as console drivers.



5. Software

The software folder provides a platform for developing various applications, independent of the underlying BSP. All the applications/projects developed in SHAKTI-SDK reside in this folder. In general, an application will involve writing high level C/C++ code that uses BSP API's. The software folder is broadly classified into three sub-directories,

5.1 Projects

This folder consists of applications developed using different sensors. These are usually a combination of standalone applications.

5.2 Benchmarking

Applications or bare metal codes that are developed for bench-marking a core reside here. These programs usually describe the capability of the SHAKTI class of processors.

5.3 Examples

This is the place where any new standalone application is developed. Few example programs involving sensors are already developed for different peripherals and kept here. These programs demonstrate the integration of BSP and the core support libraries with the user programs.

6. Setting up the SHAKTI-SDK

6.1 Download the SHAKTI-SDK repository

SHAKTI-SDK repository contains scripts, board support packages to build your application. It can be cloned by running the following command:

```
git clone https://gitlab.com/shaktiproject/software/shakti-sdk.git
```

As discussed earlier, SHAKTI-SDK helps in developing applications for the SHAKTI class of processors. We are listing the steps to develop a small application using SHAKTI-SDK.

make sure that the prerequisites mentioned below are ready.
Board is up and running with SHAKTI.
Downloaded or cloned a new SHAKTI-SDK.

6.2 Steps to add a new application to SHAKTI-SDK

The new application is created under one of the example/XYZ_applns folder.
XYZ should be a device type in the SoC. For example it can be UART, I2C, etc...
Let's assume, we are under the XYZ_applns directory.
Create a folder for the new application and name it accordingly.
Inside the folder, create source and header files for the application.• Create and edit a new Makefile for the application (refer existing examples).
The name of the application folder corresponds to the name of the application.**[NOTE:APPLICATION FOLDER NAME AND SOURCE FILE NAME SHOULD BE THE SAME]**
Make an entry in the existing Makefile under ./shakti-sdk/software/examples for the new application*.
Now typing make list-applns will list the new application as one under SHAKTI-SDK.

6.2.1 My first program !

Steps to compile and run hello world on shakti using shakti-sdk

The device required is UART. Include UART device headers for the program.
Write your program under software/examples/uart_applns/.
Create a folder called 'first'.
Create a first.c file and write a program to print "Hello World !".(refer hello code in uart applications folder)
Create and edit a Makefile for the program and save in the 'first' folder. Use existing programs in the example folder for reference.
Make a new entry for the program in the 'existing Makefile' under examples folder **[located at ./shakti-sdk/software/examples.]**
[NOTE: FOLDER NAME AND FILE NAME SHOULD BE THE SAME]

6.2.2 Build

The make commands in SHAKTI-SDK give various options to build and run a program. The list of make commands can be found by typing make help in the terminal. Once the program is built using the MAKE command, the ELF file is generated. The ELF file is the final executable that can be loaded into the memory and run.

```
$ cd shakti-sdk
$ make software PROGRAM=hello TARGET=hackathon
```

Interpreting above commands:
PROGRAM is the new one created. It is listed by typing "make list_applns"

7. Steps to run

7.1 Prerequisites

7.1.1 MINITERM WINDOW (for console output)

1. In the first terminal, open a serial port to display output from UART.

```
$ sudo miniterm.py /dev/ttyUSB0 19200
```

7.1.2 OPENOCD WINDOW

In the second terminal, launch OpenOCD with super user (sudo) permission. Please ensure that you are in the SHAKTI-SDK folder.

```
$ pwd
/home/user/shakti-sdk ———-> you are in the right folder
Press reset in the board and run the following commands.
$ cd ./bsp/third_party/artix7_35t
$ sudo $(which openocd) -f ftdi.cfg
```

7.1.3 GDB WINDOW

3. In the third terminal launch RISC-V GDB. Applications will be loaded and run in this terminal.

7.1.4 BUILD WINDOW

In the fourth terminal launch RISC-V GDB. Applications will be loaded and run in this terminal.

[NOTE:Please ensure that you are in the SHAKTI-SDK folder.]

```
$ make software PROGRAM=hello TARGET=hackathon
[NOTE:here hello is my application i am running please replace it with your application name]
```

Note:

1. "/dev/ttyUSB0" - ttyUSB means "USB serial port adapter" and the "0" ("0" or "1" or whatever) is the USB device number.
2. The default baud rate is 19200.