Getting started with EZ-USB™ FX20/FX10/FX5N/FX5

About this document

Scope and purpose

This application note introduces the capabilities and features of the EZ-USB™ FX20 USB 20 Gbps device controller. This application note also helps to get started with the product design using FX20. Additionally, it provides an overview of FX20 development tools, hardware kits, and other useful resources.

Intended audience

This document is primarily intended for anyone who uses FX20 for designing USB 20 Gbps devices.

Note: This document focuses mostly on FX20 devices, but is also applicable for FX10, FX5N, and FX5 devices. See the relevant document references for your device.

Table of contents

About this document **1**
Table of contents **2**
1 Introduction **4**
1.1 FX20 family general description and comparison 4
1.2 Block diagram 5
1.3 FX20 vs FX3 specifications 6
2 Features **7**
2.1 High bandwidth data subsystem (HBWSS) 7
2.1.1 USB 3.2 Gen 2x2 peripheral interface 7
2.1.2 High bandwidth data interface 7
2.1.2.1 LVDS receiver interface 7
2.1.2.2 LVCMOS interface 8
2.1.3 GPIF III 10
2.1.4 High bandwidth direct memory access (HB-DMA) 10
2.2 CPU and memory subsystem 11
2.3 Fixed-function digital blocks 11
2.3.1 Serial Communication Blocks (SCBs) 11
2.3.2 Quad SPI/serial memory interface (SMIF) 12
2.3.3 Timer/counter pulse-width modulator (TCPWM) 12
2.3.4 CAN interface 12
2.3.5 USB Full-Speed device interface 12
2.4 GPIOs 13
2.5 Special-function peripherals 13
2.5.1 Audio subsystem 13
2.6 SWD/JTAG interface 14
2.7 USB Type-C orientation detection and correction 14
3 USB 3.2 overview **15**
3.1 Differences between FX20, FX10, FX5N, and FX5 16
3.2 USB references 16
4 HBWSS terminologies **17**
4.1 LVDS/LVCMOS ports vs. LVDS/LVCMOS links 17
4.1.1 Link calibration (LVDS/LVCMOS) configuration 17
4.1.1.1 PHY and link training 17
4.1.1.2 Only link training 17
4.1.1.3 No PHY or Link training 17
4.2 GPIF III threads and sockets 17
5 Development tools for FX20 ..... **19**
5.1 ModusToolbox™ 19
5.1.1 Support for other IDEs 19
5.1.2 Programming and debugging 19
5.1.3 Toolchains 19
5.2 EZ-USB™ FX Control Center 19
5.3 EZ-USB™ FX Code Builder and Configurator 20
5.4 USB drivers 20
5.5 Useful hardware tools and software 21
5.5.1 USB protocol analyzer 21
5.5.2 Logic analyzer 21

Table of contents

5.5.3 USBView 21
5.5.4 UVC host applications 22
5.5.5 USB3 Vision host applications 23
5.5.6 Serial terminal (Tera Term, Cool Term, etc.,) 23
6 Introduction to the FX20 hardware kit 24
6.1.1 FX20 development kit (KIT_FX20_FMC_001) 24
7 Firmware stack 25
7.1.1 USB bootloader 25
7.1.2 FX20 code examples 26
8 Getting started with FX20 application examples 27
8.1 Hello World 29
8.2 UVC-UAC 30
9 Migrating the FX3 design to FX5 or FX20 32
9.1 Migrating FX3-based hardware design to FX20 32
9.1.1 Power supply and I/O voltage comparison 32
9.1.2 Crystal/clock 32
9.2 Migrating FX3-based firmware application to FX20 32
References 34
Glossary 36
Revision history 38
Disclaimer 39

1 Introduction

1.1 FX20 family general description and comparison

FX20 is a family of USB 20 Gbps (USB 3.2 Gen 2x2) device controllers targeting emerging applications in the video, audio, and other generic data acquisition markets. FX20 contains Cortex®-M4 and M0+ MCUs, 512 KB flash, 128 KB SRAM, 128 KB ROM, Serial Communication Blocks (SCBs), and a cryptographic engine to support various security features. A high bandwidth data subsystem provides DMA-based data transfers between low-voltage differential signaling (LVDS)/low-voltage CMOS (LVCMOS) interface and USB ports at speeds up to 20 Gbps. An additional 1-MB SRAM is included in the high bandwidth data subsystem to provide buffering for USB data. FX20 also supports USB Type-C plug orientation detection and flip-mux function without needing an external high-speed mux.

FX20 provides a USB 20 Gbps upgrade to the FX3 family that has been widely adopted in the camera, video, and imaging applications. FX20 provides up to 6x bandwidth increase over the FX3 and helps unleash the new capabilities of latest image sensors and ultra-high-definition videos, delivering higher resolution, higher frame rate and deeper color images over USB 20 Gbps.

Table 1 shows FX20, FX10, FX5N and FX5 device variants. This table can be used to select the suitable device for the required application. USB speed support is the only difference between FX20 and other devices. FX5 only supports USB speeds up to 5 Gbps. FX5N and FX10 supports USB speeds up to 10 Gbps.

Table 1 Device variants of the FX20 family

Device	USB 3.2	USB 5 Gbps	USB 10 Gbps	USB 20 Gbps	Type-C support	Maximum flash size	RAM	LVDS/LVCMOS interface
FX5	Gen 1	X	-	-	X	512 KB	1 MB	X
FX5N	Gen 1x2	X	X	-	X	512 KB	1 MB	X
FX10	Gen 2	X	X	-	X	512 KB	1 MB	X
FX20	Gen 2x2	X	X	X	X	512 KB	1 MB	X

1.2 Block diagram

The following figure (Figure 1) shows the FX20 block diagram overview. See the datasheet for a detailed block diagram. This block diagram is applicable for FX10, FX5N, and FX5 devices as well. The only difference is USB bandwidth of USB device controller block.

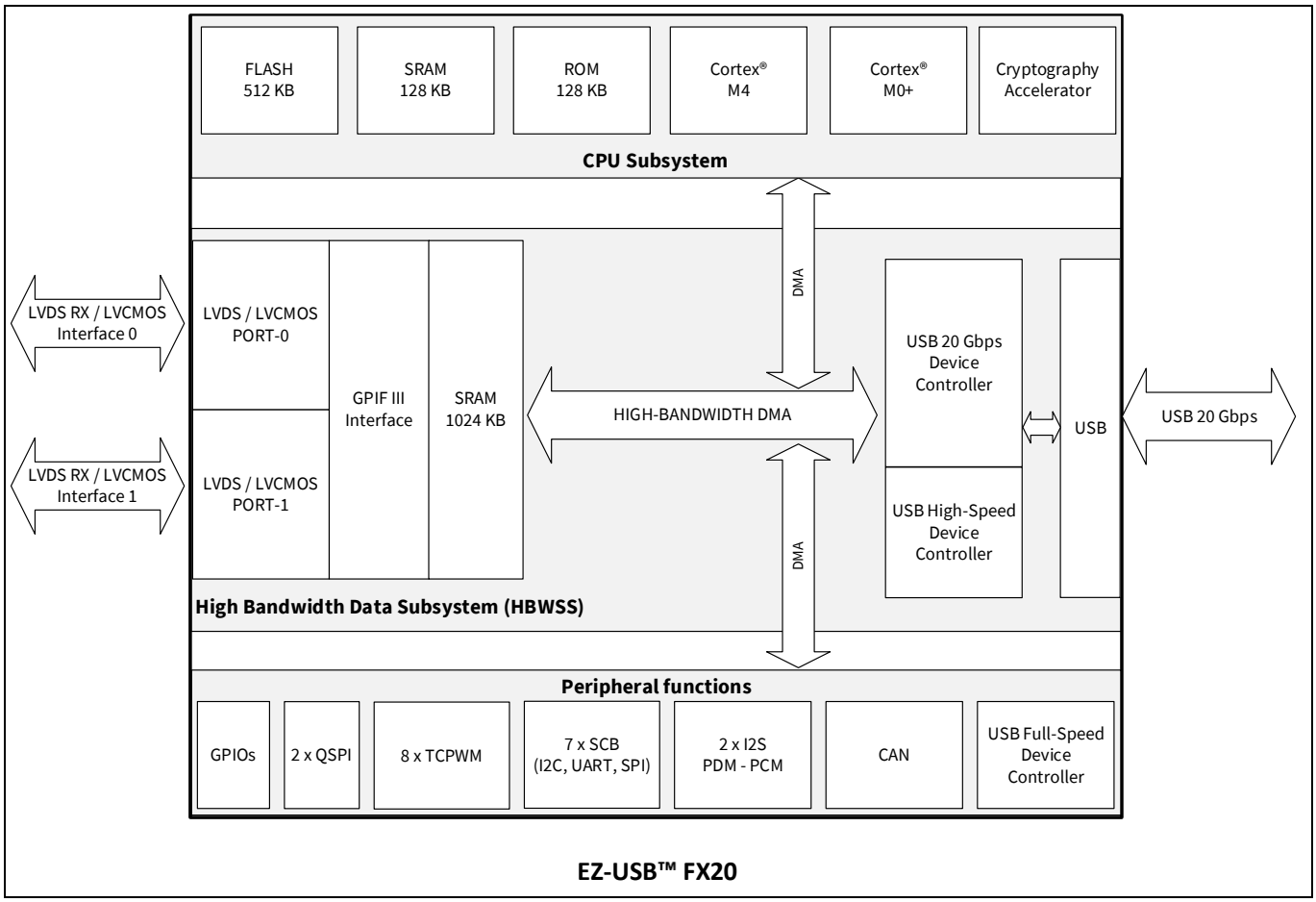

Figure 1 FX20 block diagram overview

1.3 FX20 vs FX3 specifications

Table 2 shows the differences between FX20, FX10, FX5N, FX5, FX3, and CX3.

Table 2 Specifications comparison table

Features	FX3	CX3	FX5	FX5N	FX10	FX20
USB specification (Bus speed)	USB 5 Gbps			USB 10 Gbps		USB 20 Gbps
FPGA/image sensor/ ISP interface	Parallel 32-bit GPIF II (100 MHz)	4 lanes MIPI CSI-2 (1 Gbps/lane)	16-lane LVDS (up to 1.25 Gbps/lane) 32-bit LVCMOS (160 MHz DDR & SDR)			
Interface configurability	FIFO, SRAM, A/D Mux, etc.	2/4-lane MIPI RX	LVDS (RX only) - 8-lane x2 LVDS / subLVDS LVCMOS (DDR & SDR): 32-bit Tx/Rx			
Device Data throughput	~3.2 Gbps	~2.4 Gbps	~4 Gbps	~8 Gbps	~9.6 Gbps	~19.2 Gbps
USB device	USB 5 Gbps device	USB 5 Gbps device	USB 5 Gbps device and	USB 10 Gbps device	USB 10 Gbps device	USB 20 Gbps device
Processor	ARM9		Arm® Cortex® M4F and M0+			
In-built Flash	No		Up to 512 KB			
SRAM	512 KB		128 KB + 1024 KB			
USB Type-C orientation detection and correction	No		Yes (It has built-in detection; external MUX is not required to detect USB Type-C orientation)			
Security (Cryptographic engine)	No		Yes (AES, DES, SHA, RSA)			
Serial interfaces	1x SPI/I2C/UART		7x I2C/SPI/UART 2x I2S (supports PDM) 2x QSPI			
Other Interfaces	-NA-		USB Full-Speed device			
Boot Modes supported	USB, SPI Flash, I2C EEPROM, Async ADMux		USB, Internal Flash			
Software development kit	FX3 SDK		ModusToolbox™			
Packages	121-BGA (10 x 10 mm, 0.8 mm pitch)		169-BGA (10 x 10 mm, 0.75 mm pitch)			

2 Features

2.1 High bandwidth data subsystem (HBWSS)

The high bandwidth data subsystem provides data transfers over DMA between the LVDS/LVCMOS interface and the USB 3.2 Gen 2x2 interface at speed up to 20 Gbps. The 1024 KB SRAM is included in this subsystem to provide sufficient buffering for data. This subsystem is also interfaced with the peripheral and system interconnect for data transfers to other low-bandwidth peripherals.

2.1.1 USB 3.2 Gen 2x2 peripheral interface

The high bandwidth data subsystem consists of a USB 3.2 Gen 2x2 peripheral interface with the following features:

- Configurable to operate in USB High-Speed (480 Mbps), USB 3.2 Gen 1 (5 Gbps), USB 3.2 Gen 1x2 (10 Gbps), USB 3.2 Gen 2 (10 Gbps), and USB 3.2 Gen 2x2 (20 Gbps)
- Up to 32 endpoints (EPs): control EP IN/OUT, 15 IN, 15 OUT; each function endpoint (EP) can be configured as bulk, isochronous, or interrupt type
- Supports simultaneous transactions across multiple IN EPs
- Supports all USB low-power states
- Supports UVC, UAC, HID, CDC, U3V, and vendor class protocols
- Supports USB precision time measurement (PTM)
- Supports USB bus-powered and self-powered mode
- Detects USB-C connector orientation using CC lines. External Rd resistors are required on CC lines

2.1.2 High bandwidth data interface

High bandwidth data interface can be configured as LVDS or LVCMOS modes. It has two ports, and each port can be configured independently to support the LVDS receiver interface or LVCMOS receiver/transmitter interfaces.

2.1.2.1 LVDS receiver interface

The LVDS ports can be configured to support one of the following modes:

- Two narrow links with 1, 2, 4, or 8 data lanes, one control lane, one clock lane and 7 LVCMOS control signals
- The LVDS interface supports clock frequencies from 74.25 MHz to 625 MHz[1]
- The gearing ratios[2] 1:1, 2:1, 4:1, and 8:1 is supported on each configuration of the link
- Maximum data rate per lane is 1250 Mbps (e.g., 625 MHz clock and 2:1 gearing ratio)
- Each LVDS link has 7 LVCMOS control signals of which 3 are input only and the other 4 are input/output control signals
- One wide link with up to 16 data lanes, one control lane, and two clock lanes. The same clock must be fed to both clock lanes in this mode as only GPIF III state machine 1 is active in this mode.

[1] LVDS interface clock frequency shall not exceed the limit with selected gearing ratio.

[2] Gearing ratio indicates the number of bits transferred, per data lane, per clock on the interface. For e.g., a gearing ratio of 8:1 means that 8 bits of data are transferred per lane in each clock.

Features

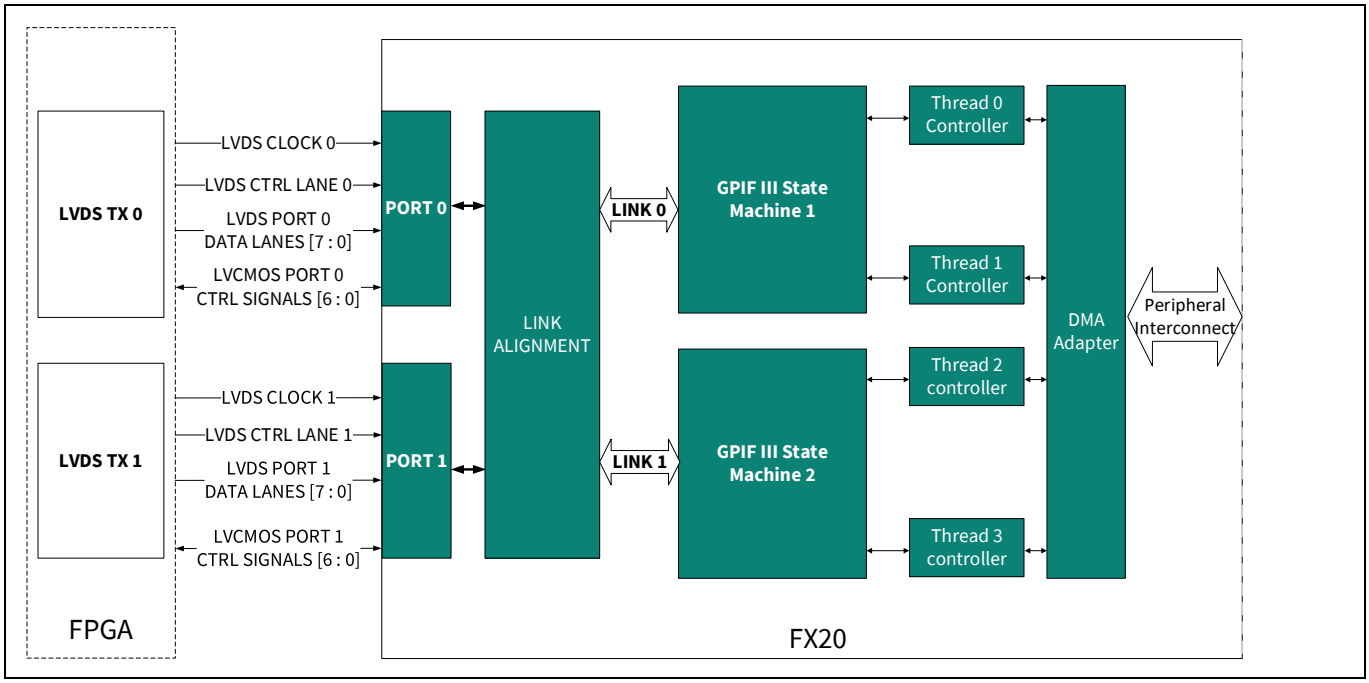

Figure 2 LVDS narrow link configuration

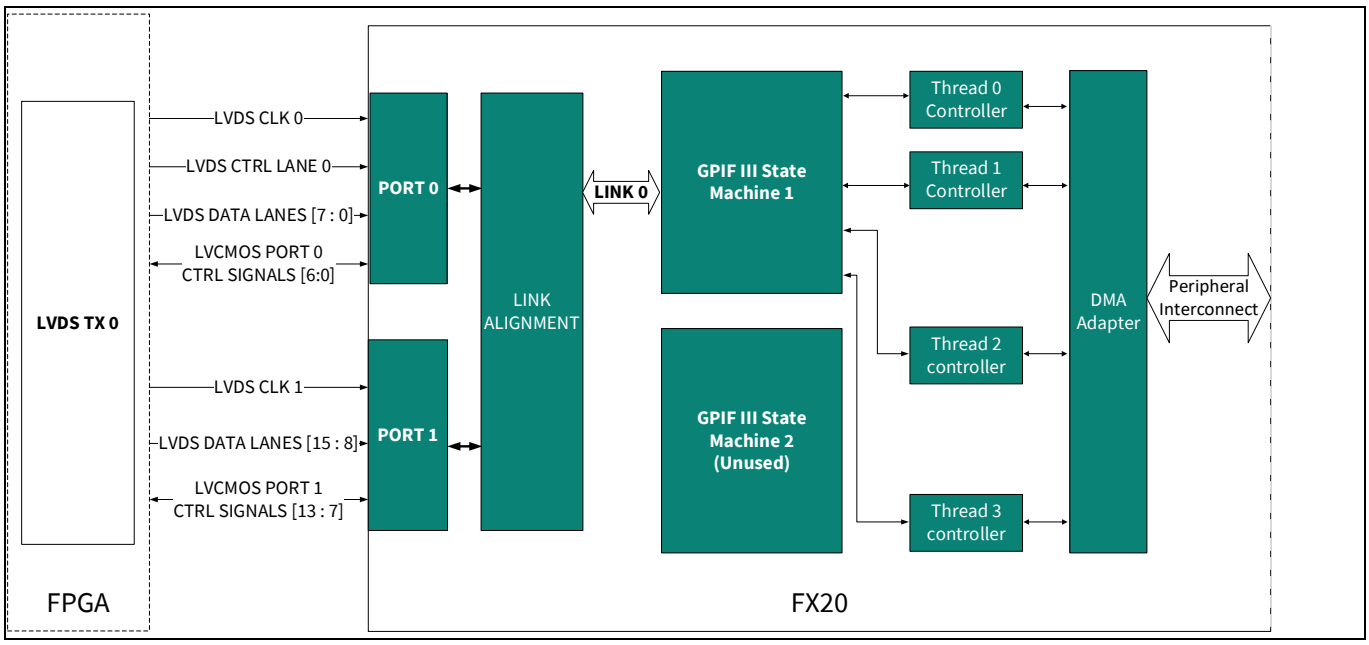

Figure 3 LVDS wide link configuration

2.1.2.2 LVCMOS interface

The LVCMOS parallel interface can support both Receiver and Transmitter modes of operation. Each port contains 16 LVCMOS I/Os and 10 control signals. The LVCMOS receiver interface is backward compatible with the GPIF-II interface in FX3.

The LVCMOS Interface can also support similar link configurations as the LVDS RX Interface.

Features

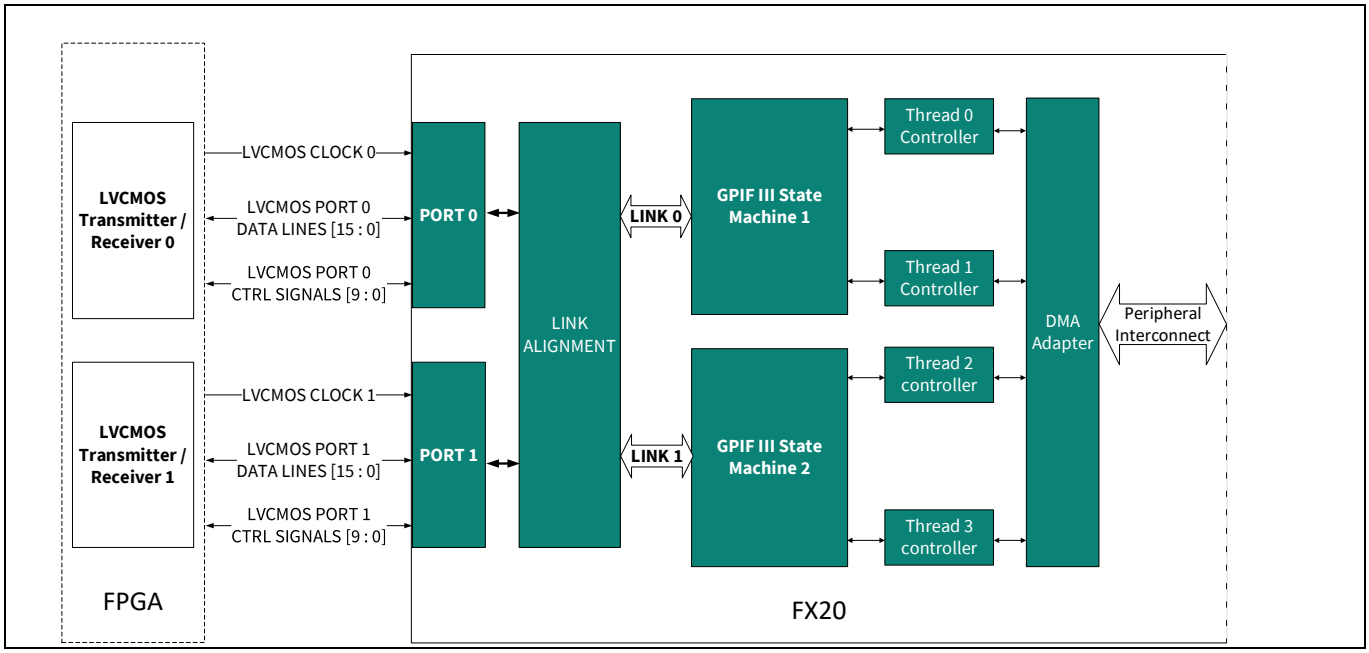

Figure 4 LVCMOS narrow link configuration

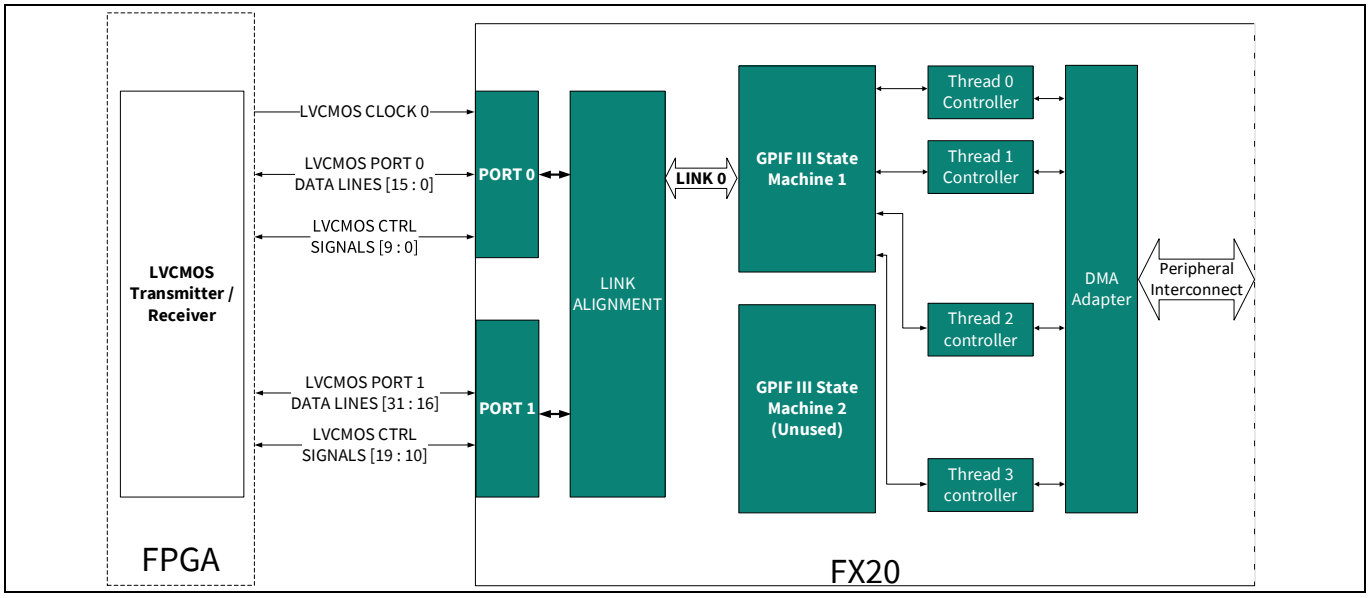

Figure 5 LVCMOS wide link configuration

The maximum data rate on each LVCMOS IO is:

- 160 MHz for single data rate (SDR) and double data rate (DDR) in Receiver mode
- 100 MHz for SDR and 80 MHz for DDR in Transmitter mode [DDR TX works in Master Mode only]

Note: FX20 can also support a combination of LVDS and LVCMOS interfaces as well on its ports, i.e., Port 0 can be in LVDS Receiver mode and Port 1 in LVCMOS mode or vice-versa.

2.1.3 GPIF III

The GPIF III is a programmable state machine that enables a flexible interface that may function as a master or slave in industry-standard or proprietary interfaces. This interface, along with the mechanism to route the received data to the DMA, forms the link layer of the HBWSS.

The following are the GPIF III features:

- Functions as master or slave
- Provides 256 firmware programmable states
- Supports 8-bit, 16-bit, 24-bit, and 32-bit parallel data bus
- Supports two independent instances of GPIF III state machines, each connected to a link or a single GPIF III state machine (GPIF III state machine 1) connected to a wide link

The GPIF III state transitions are based on control input signals. The control output signals are driven due to the GPIF III state transitions. For more details, contact Infineon support.

2.1.4 High bandwidth direct memory access (HB-DMA)

The HB-DMA controller handles data transfer within the HBWSS and with other peripherals of the FX20. It has the following features:

- Supports multiple simultaneous streams
- Supports maximum throughput of 20 Gbps
- DMA is configurable by the CPU. After configuration, the DMA operates automatically and transfers the data stream from the LVDS/LVCMOS interface to the USB 3.2 interface without CPU intervention
- It can add headers and footers to each data payload without CPU intervention (insert metadata feature)

2.2 CPU and memory subsystem

The FX20 CPU subsystem has the following major components:

- 150 MHz Arm® Cortex®-M4 (CM4) CPU with single-cycle multiply, floating point, and memory protection unit (MPU)
- Up to 100 MHz Arm® Cortex®-M0+ (CM0+) CPU with single-cycle multiply and MPU, however, for CM4 speeds above 100 MHz, CM0+ and bus peripherals are limited to half the speed of CM4
- 512 KB application flash with read-while-write (RWW) support
- Two 8 KB flash caches, one for each CPU
- 128 KB SRAM with power and data retention control
- 1 KB one-time programmable (OTP) eFuse array
- DMA
- Cryptography accelerator

Generally, all memory and peripherals can be accessed and shared by all bus masters through the multi-layer Arm® AMBA high-performance bus (AHB) arbitration. Accesses between CPUs are synchronized using an inter-processor communication (IPC) block.

2.3 Fixed-function digital blocks

2.3.1 Serial Communication Blocks (SCBs)

FX20 has seven SCBs. These SCBs can be configured in the following modes:

- **I2C mode:** The SCB can implement a full multi-master and slave interface (it is capable of multi-master arbitration). This block can operate up to 1 Mbps (Fast Mode Plus). It also supports EZI2C, which creates a mailbox address range and effectively reduces I2C communication to reading from and writing to an array in memory. The SCB supports a 256-byte FIFO for receiving and transmitting. The I2C peripheral is compatible with I2C Standard mode, Fast mode, and Fast mode Plus devices as defined in the NXP I2C-bus specification and user manual (UM1024). The I2C bus I/O is implemented with GPIO in Open-drain modes.
- **UART mode:** This full-featured UART operates at up to 8 Mbps. It supports automotive single-wire interface (LIN), infrared interface (IrDA), and SmartCard (ISO 7816) protocols, all of which are minor variants of the basic UART protocol. In addition, it supports the 9-bit Multiprocessor mode that allows the addressing of peripherals connected over common Rx and Tx lines. Common UART functions such as parity error, break detection, and frame error are supported. A 256-byte FIFO allows much greater CPU service latencies to be tolerated.
- **SPI mode:** The SPI mode supports full Motorola SPI, TI Secure Simple Pairing (SSP) (which essentially adds a start pulse that is used to synchronize SPI Codecs), and National Microwire (a half-duplex form of SPI). The SPI block supports an EZSPI mode in which the data interchange is reduced to reading and writing an array in memory. The SPI interface operates with a 25-MHz clock.

See the FX20 datasheet (contact Infineon support) for more details on modes supported on each SCB.

2.3.2 Quad SPI/serial memory interface (SMIF)

FX20's serial memory interface can run up to 80 MHz. It supports single, dual, quad, dual-quad, and octal SPI configurations and supports up to four external memory devices. It supports two modes of operation:

- **Memory mapped input output (MMIO):** A command mode interface that provides data access via registers and FIFOs
- **Execute-in-place (XIP):** In which CPU reads and writes are directly translated to SPI read and write transfers

In XIP mode, external memory is mapped into the FX20 internal address space, enabling code execution directly from the external memory. To improve performance, a 4-KB cache is included. XIP mode also supports AES-128 on-the-fly encryption and decryption, enabling secured storage and access of code and data in the external memory.

2.3.3 Timer/counter pulse-width modulator (TCPWM)

FX20 has eight 32-bit TCPWM blocks. They have the following features:

- Each TCPWM block supports the following operational modes:
  - Timer-counter with compare
  - Timer-counter with capture
  - Quadrature decoding
  - Pulse-width modulation (PWM)
  - Pseudo-random PWM
  - PWM with dead-time
- Up, down, and up/down counting modes
- Clock prescaling (division by 1, 2, 4... 64, 128)
- Double buffering of compare/capture and period values
- Underflow, overflow, and capture/compare output signals
- Supports interrupt on:
  - Terminal count – Depends on the mode; typically occurs on overflow or underflow
  - Capture/compare – The count is captured to the capture register, or the counter value equals the value in the compare register
- Complementary output for PWMs
- Selectable start, reload, stop, count, and capture event signals for each TCPWM; with the rising edge, falling edge, both edges, and level trigger options. The TCPWM has a Kill input to force outputs to a predetermined state

2.3.4 CAN interface

FX20 has one CAN FD block for industrial and automotive applications. The block includes time-stamp support and has a 4-KB message RAM. FD data rates of up to 5 Mbps are supported. DMA transfers are supported.

2.3.5 USB Full-Speed device interface

In addition to the USB interface block that supports up to USB 3.2 Gen 2x2 device speeds, FX20 also has an independent USB Full-Speed (12 Mbps) device interface. This interface can be connected to a separate USB connector with the USBFSDN and USBFSDP pins.

The device can have up to eight endpoints. A 512-byte SRAM buffer is provided with DMA supported.

In FX20 code examples, this interface is commonly used to view debug messages over the USB CDC interface.

2.4 GPIOs

FX20 has 48 GPIOs including the pins used for low-speed peripheral interface. It supports the following features:

- Eight drive modes:
  - Input only
  - Weak pull-up with strong pull-down
  - Strong pull-up with weak pull-down
  - Open drain with strong pull-down
  - Open drain with strong pull-up
  - Strong pull-up with strong pull-down
  - Weak pull-up with weak pull-down
- In Strong Drive mode, there are four selectable drive strengths for IOL and IOH current drive
- Input thresholds select (CMOS or LVTTL)
- Selectable slew rates for dV/dt-related noise control to improve EMI
- The pins are organized into logical entities called ports, which are up to eight pins wide. Data output and pin state registers store, respectively, the values to be driven on the pins and the input states of the pins.
- Every pin can generate an interrupt if enabled and each port has an interrupt request (IRQ) associated with it
- The ports 10.0 and 10.1 pins are capable of overvoltage tolerant (OVT) operation, where the input voltage may be higher than VDDD. OVT pins are commonly used with I2C to allow powering the chip OFF while maintaining a physical connection to an operating I2C bus without affecting its functionality.
- GPIO pins can be ganged to source or sink higher values of current

2.5 Special-function peripherals

2.5.1 Audio subsystem

This subsystem consists of the following hardware blocks:

- Two Inter-IC Sound (I2S) interfaces for audio input and output
- Two pulse density modulation (PDM) to pulse code modulation (PCM) decoder channels for audio input

Each of the I2S interfaces implements two independent hardware FIFO buffers – TX and RX, which can operate in master or slave mode. The following features are supported:

- Multiple data formats – I2S, left-justified, Time-division multiplexed (TDM) mode A, and TDM mode B
- Programmable channel/word lengths – 8/16/18/20/24/32 bits
- Internal/external clock operation up to 192 ksps
- Interrupt mask events – trigger, not empty, full, overflow, underflow, and watchdog
- Configurable FIFO trigger level with DMA support

The I2S interface is commonly used to connect with audio codecs, simple DACs, and digital microphones.

Features

The PDM-to-PCM decoder implements a single hardware RX FIFO that decodes a stereo or mono 1-bit PDM input stream to PCM data output. The following features are supported:

- Programmable data output word length – 16/18/20/24 bits
- Programmable gain amplifier (PGA) for volume control from -12 dB to +10.5 dB in 1.5-dB steps
- Configurable PDM clock generation. Range from 384 kHz to 3.072 MHz
- Droop correction and configurable decimation rate for sampling; up to 48 ksps
- Programmable high-pass filter gain
- Interrupt mask events – not empty, overflow, trigger, and underflow
- Configurable FIFO trigger level with DMA support

The PDM-to-PCM decoder is commonly used to connect to digital PDM microphones. Up to two microphones can be connected to the same PDM Data line.

2.6 SWD/JTAG interface

FX20 has a debug access port (DAP) that acts as the interface for device programming and debugging.

An external programmer or debugger (the “host”) communicates with the DAP through the device Serial Wire Debug (SWD) or Joint Test Action Group (JTAG) interface pins. Through the DAP (and subject to restrictions), the host can access the device memory and peripherals as well as the registers in both CPUs.

Each CPU offers the following debug and trace features:

- CM4 supports:
  - Six hardware breakpoints and four watchpoints
  - 4-bit embedded trace macrocell (ETM)
  - Serial wire viewer (SWV)
  - `printf()`-style debugging through the single wire output (SWO) pin.
- CM0+ supports
  - Four hardware breakpoints and two watchpoints
  - A micro trace buffer (MTB) with 4-KB dedicated RAM
  - Additionally, FX20 has an embedded cross-trigger for synchronized debugging and tracing of both CPUs

2.7 USB Type-C orientation detection and correction

FX20 has support of detecting USB Type-C connection orientation and correction. It does not require external multiplexer to detect and correct.

3 USB 3.2 overview

USB 20 Gbps or USB 3.2 Gen 2x2 specification is primarily a performance enhancement to USB 5 Gbps to provide more bandwidth for devices such as Solid-State Drives (SSD) and High-Definition (HD) displays by adding dual-lane support to the USB Type-C cable and connector. The goal remains the same; end users view these the same as they viewed USB 2.0 and USB 3.0, just higher in performance.

USB 3.2 is similar to earlier versions of USB in that it is a cable bus supporting data exchange between a host computer and a wide range of simultaneously accessible peripherals. The attached peripherals share bandwidth through a host-scheduled protocol. The bus allows peripherals to be attached, configured, used, and detached while the host and other peripherals are in operation.

USB 3.2 is a dual-bus architecture that provides backward compatibility with USB 2.0. One bus is a USB 2.0 bus (see Universal Serial Bus Specification, Revision 2.0) and the other is an Enhanced 20 Gbps. USB 3.2 specifically adds dual-lane support.

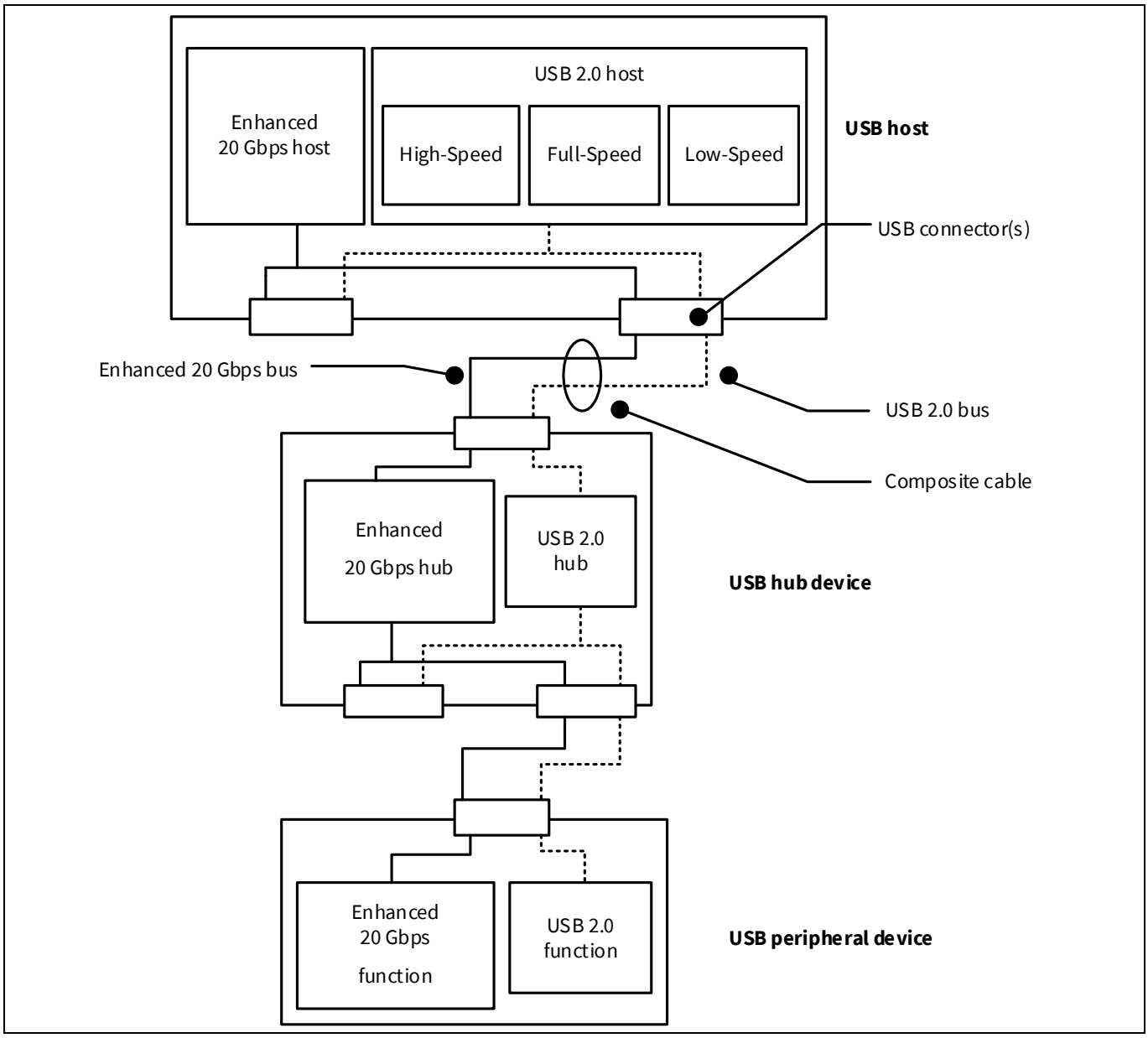

Figure 6 USB 3.2 dual bus system architecture

3.1 Differences between FX20, FX10, FX5N, and FX5

USB 3.2 Gen 1 and USB 3.2 Gen 2 are two different generations of the USB 3.2 standard. The main difference between the two is their maximum theoretical transfer speed. USB 3.2 Gen 1 has a maximum theoretical transfer speed of 5 Gbps, while USB 3.2 Gen 2 has a maximum theoretical transfer speed of 10 Gbps. This means that USB 3.2 Gen 2 is two times as fast as USB 3.2 Gen 1. USB 3.2 Gen 1x2 uses two data lanes of 5 Gbps (USB 3.2 Gen 1) and achieves a total speed of up to 10 Gbps. Whereas USB 3.2 Gen 2x2 uses two data lanes of 10 Gbps (USB 3.2 Gen 2) and achieves a total speed of up to 20 Gbps.

There is also a difference in data encoding between USB 3.2 Gen 1 and USB 3.2 Gen 2. USB 3.2 Gen 1 uses 8b/10b encoding, while USB 3.2 Gen 2 uses 128b/132b encoding. This means that USB 3.2 Gen 2 can transfer more data in a single transmission which results in faster transfer speeds.

In 8b/10b encoding, each 8-bit data byte is encoded as a 10-bit signal. This means that there is a 20% overhead, as only 80% of the signal is actual data. In 128b/132b encoding, each 128-bit data byte is encoded as a 132-bit signal. This means that there is only a 3% overhead, as 97% of the signal is actual data.

As a result of the more efficient data encoding, USB 3.2 Gen 2 can achieve theoretical transfer speeds of up to 10 Gbps, while USB 3.2 Gen 1 can only achieve theoretical transfer speeds of up to 5 Gbps.

In practice, the actual transfer speeds of USB 3.2 Gen 1 and USB 3.2 Gen 2 devices will be lower than the theoretical speeds. However, USB 3.2 Gen 2 devices should still be able to achieve significantly faster transfer speeds than USB 3.2 Gen 1 devices.

FX5 is a USB 3.2 Gen 1 device and uses a single lane of 5 Gbps to achieve up to 5 Gbps speed and FX5N is a USB 3.2 Gen 1x2 device and uses a dual lane of 5 Gbps to achieve up to 10 Gbps speed.

FX10 is a USB 3.2 Gen 2 device and uses a single lane of 10 Gbps to achieve up to 10 Gbps speed and FX20 is a USB 3.2 Gen 2x2 device and uses a dual lane of 10 Gbps to achieve up to 20 Gbps speed.

3.2 USB references

- USB 3.2 Specification
- USB Design By Example, John Hyde
- USB 3.0 Technology, Mindshare

4 HBWSS terminologies

4.1 LVDS/LVCMOS ports vs. LVDS/LVCMOS links

The port terminology refers to the physical pins of the LVDS/LVCMOS I/O interface. These pins are named with a prefix of the port number, i.e., P0CP or P1CN, where P0 stands for Port 0 and P1 stands for Port 1, respectively.

The data flow path between the ports and the GPIF III interface is called links.

In FX20, two ports can be configured to be independent data paths (or links), or two ports can be merged to form a single link to route data from two ports to the GPIF III interface as a single link (wide link).

4.1.1 Link calibration (LVDS/LVCMOS) configuration

FX20 requires training for calibration LVDS/LVCMOS interface.

4.1.1.1 PHY and link training

Both PHY and Link training required when interface is configured as below:

- LVDS Wide Link
- LVDS Narrow Link Port0
- LVDS Narrow Link Port1

4.1.1.2 Only link training

Only Link training required when interface is configured as below:

- LVCMOS DDR Wide Link

4.1.1.3 No PHY or Link training

PHY or Link training NOT required when interface is configured as below:

- LVCMOS SDR Wide Link
- LVCMOS SDR Narrow Link Port0
- LVCMOS SDR Narrow Link Port1
- LVCMOS DDR Narrow Link Port0
- LVCMOS DDR Narrow Link Port1

4.2 GPIF III threads and sockets

- **Socket:** It is a point of connection between a peripheral hardware block and the HBWSS SRAM. Each peripheral hardware block on FX20, such as USB, GPIF III, UART, SPI, I2C, has a fixed number of sockets associated with it. The number of independent data flows possible through a peripheral is equal to the number of sockets in the peripheral
- **GPIF thread:** It is a dedicated data path in the GPIF III block that connects an LVDS/LVCMOS link to a socket
- **DMA buffer:** It is a section of HBWSS SRAM used for intermediate storage of data transferred through the FX20 device. DMA buffers are allocated from the HBWSS SRAM by the FX20 firmware, and their addresses are stored as part of DMA descriptors

HBWSS terminologies

- **DMA descriptor:** It is a set of registers allocated in the HBWSS SRAM. It holds information about the address and size of a DMA buffer as well as pointers to the next DMA descriptor. These pointers create DMA descriptor chains. A socket uses the values of the DMA buffer address, DMA buffer size, and DMA descriptor chain stored in a DMA descriptor for data management
- **Producer socket:** A socket that writes data to a DMA buffer is called a producer socket
- **Consumer socket:** A socket that reads data from a DMA buffer is called a consumer socket
- **Socket-switching latency:** A socket takes a finite amount of time (up to 68 clock cycle) to switch from one DMA descriptor to another after it fills or empties a DMA buffer. The socket cannot transfer any data while this switch is in progress. This latency can be a problem for interfaces that have no flow control. This issue can be addressed by using multiple GPIF threads
- **GPIF III thread switching:** Each thread will have an associated socket and DMA buffers. The GPIF III state machine uses internal control signals or external inputs (Example- address lines A1, A0 for a 2-bit slave FIFO, or address lines A0, A1, A2, A3, and A4 for a 5-bit slave FIFO) to select the active GPIF thread and socket

  Switching the active GPIF thread switches the active socket for the data transfer, thereby changing the DMA buffer used for data transfers. The GPIF thread selection mechanism is like a mux and therefore, this switch has no latency. When one DMA buffer associated with a GPIF socket is full, you can switch the GPIF thread address and access the DMA buffer associated with another GPIF socket.

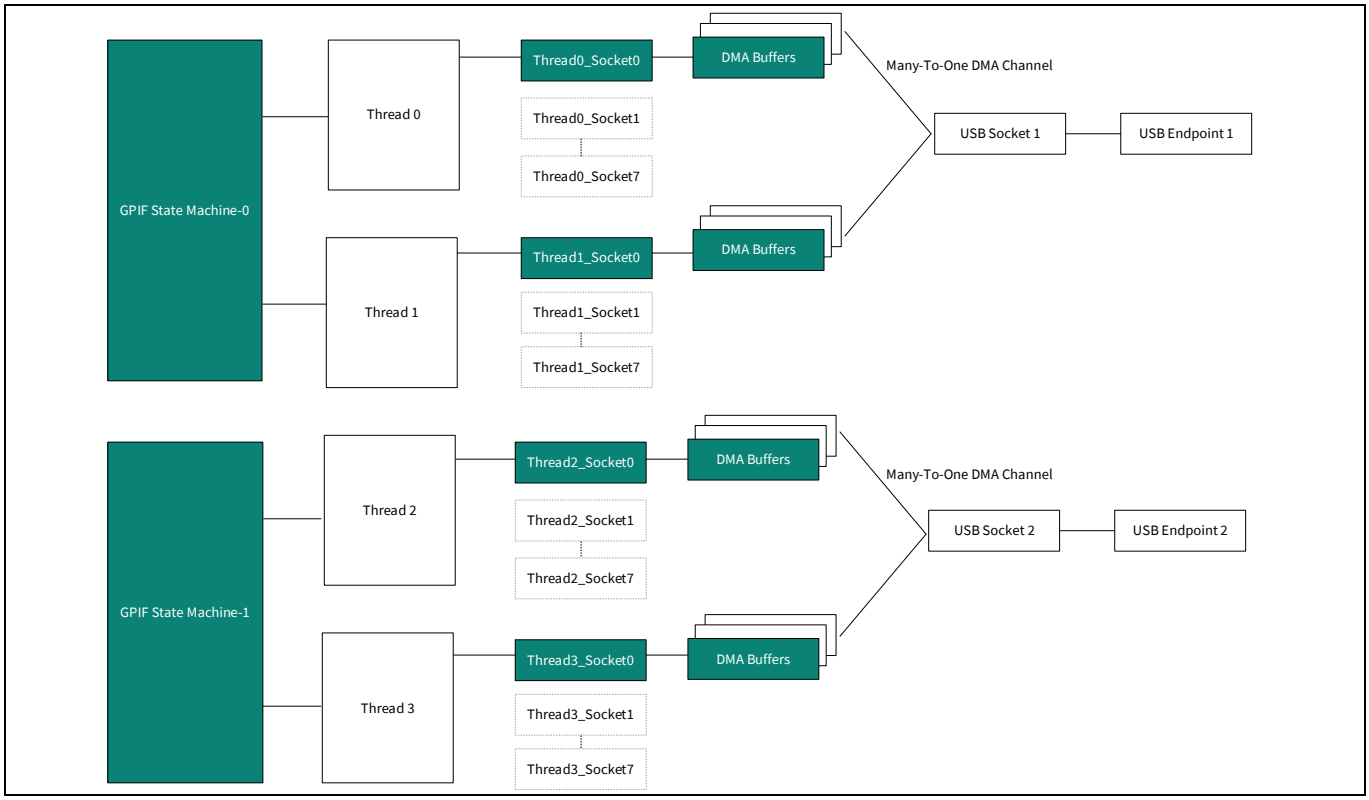

Figure 7 System that uses thread switching to maintain two independent USB data paths

See the datasheet for more details on the data flow and the metadata insertion feature in FX20.

5 Development tools for FX20

5.1 ModusToolbox™

ModusToolbox™ development platform is used for firmware/application development with the FX20. This latest-generation toolset includes the Eclipse IDE, which supports Windows, Linux, and macOS platforms. The ModusToolbox™ includes configurators, low-level drivers, middleware libraries, and other packages that can create FX20 applications. The configurators can set the configuration of different blocks in the device and generate code to use in the firmware development.

The Eclipse IDE for ModusToolbox™ is integrated with quick launchers for tools and design configurators in the Quick Panel. ModusToolbox™ supports third-party IDEs, including Visual Studio Code, Arm® MDK (µVision®), and IAR Embedded Workbench.

For an in-depth overview of the ModusToolbox™, see the ModusToolbox™ tools package user guide.

5.1.1 Support for other IDEs

The firmware for FX20 can be developed using the any IDE such as IAR Embedded Workbench or Visual Studio Code in addition to the Eclipse IDE.

See the “Exporting to IDEs” section in the ModusToolbox™ tool package user guide for more details. Infineon recommends generating the resource configurations using the configuration tools provided with ModusToolbox™.

5.1.2 Programming and debugging

The Eclipse IDE of ModusToolbox™ supports KitProg3 and uses the Open On-Chip Debugger (OpenOCD) protocol for debugging FX20 applications. It also supports GNU Debugger (GDB) debugging using industry-standard probes like the Segger J-Link.

All FX20 kits have a KitProg3 onboard programmer/debugger. It supports Cortex® Microcontroller Software Interface Standard – Debug Access Port (CMSIS-DAP). See the KitProg3 user guide for details.

For more information on programming/debugging firmware on FX20 with ModusToolbox™, see the “Program and debug support” section in the ModusToolbox™ tools package user guide.

FX20 based examples are supported in latest ModusToolbox™. See Section 7.1.2 FX20 code examples for more details on how to build and program FX20 code examples in ModusToolbox™.

5.1.3 Toolchains

FX20 supports the following toolchains:

- GNU toolchain
- ARMCC toolchain

5.2 EZ-USB™ FX Control Center

EZ-USB™ FX Control Center tool is a standalone cross-platform (Windows, Linux, and macOS) application designed to interact with FX20. This application supports the following features:

- Firmware downloads to FX20 in USB Bootloader mode
- Program download in External Flash (for FPGA programming)

- Data transfer over control, bulk, interrupt, and isochronous endpoints
- Measure USB data throughput
- Verify data loopback over selected USB endpoints

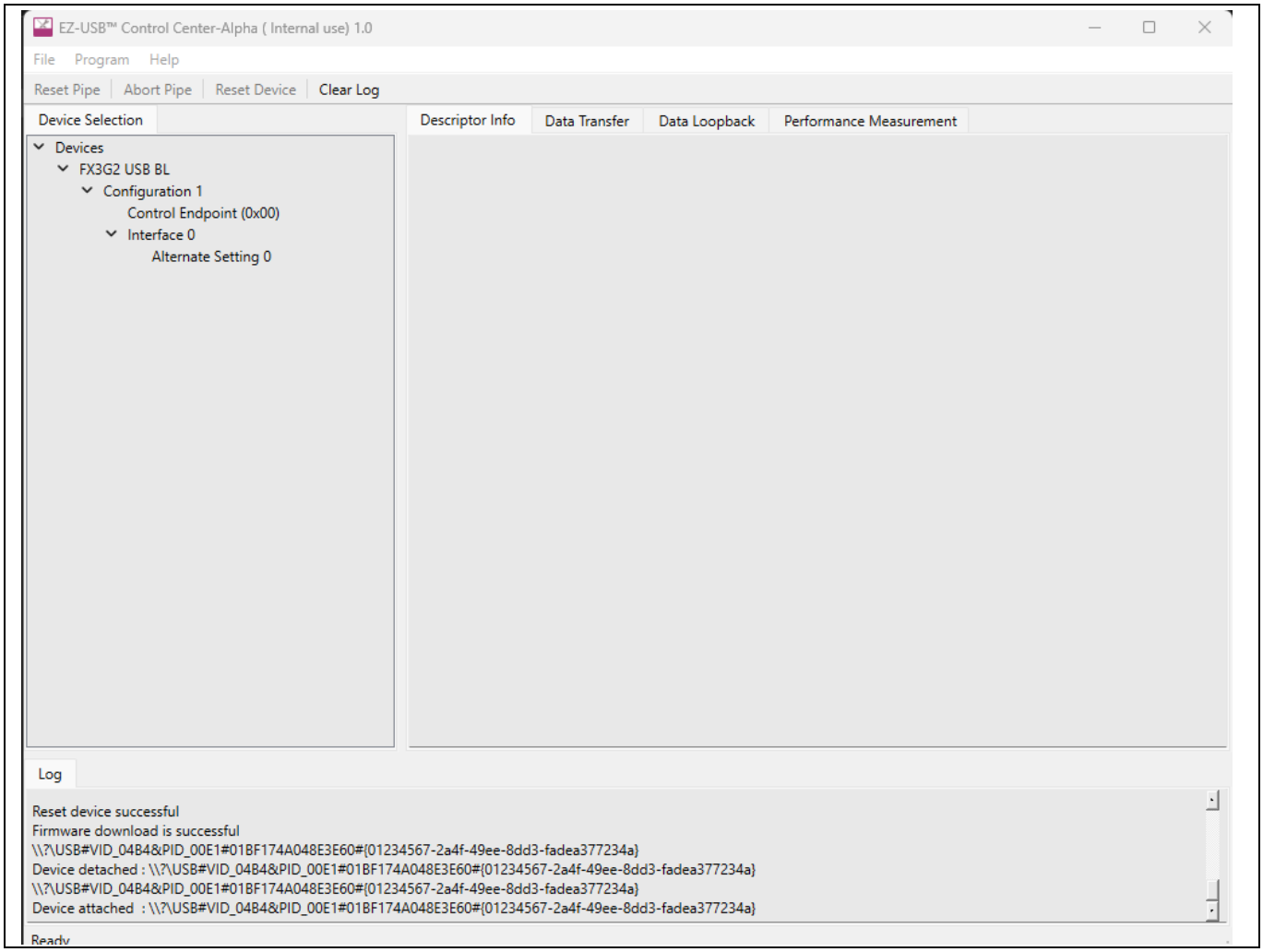

Figure 8 EZ-USB™ Control Center

EZ-USB™ FX Control Center uses WinUSB driver in Windows, and libusb in Linux and macOS platforms. You can find more details on the EZ-USB™ FX Control Center webpage.

5.3 EZ-USB™ FX Code Builder and Configurator

Contact Infineon support.

5.4 USB drivers

FX20 firmware examples support WinUSB driver by default on Windows, and libusb on Linux and macOS platforms. FX20 is also compatible with all standard-class drivers in these respective operating systems (e.g., USB Video Class, USB Audio Class, USB Mass Storage, USB Human Interface Device Class, etc.). FX20's USB3 Vision code example also supports vendor-provided USB3 Vision drivers (e.g., Pleora, A&B software, etc.)

FX20 can also support the CyUSB3 driver in Vendor Class mode.

5.5 Useful hardware tools and software

5.5.1 USB protocol analyzer

A USB protocol analyzer is a debugging tool that analyzes the traffic on the USB between FX20 and the USB host. Software tools included with each analyzer then decodes the data into USB transfer packets. By analyzing this data, issues can be easily identified, and performance can be maximized. Several USB analyzers are available in the market today. Infineon does not recommend any specific analyzer but here are a few options:

- Standalone USB protocol analyzer
  - LeCroy USB Voyager
- PC software USB 3.0 protocol analyzer
  - SourceQuest SourceUSB
  - SysNucleus USBTrace
  - Wireshark

5.5.2 Logic analyzer

Logic analyzers allow simple analysis of digital signals. They can be used to look at various signals between FX20 and other peripherals. There are two types of logic analyzers in the market:

- Standalone type logic analyzer
  - Portable Logic Analyzer
- PC-based type logic analyzer
  - Saleae Logic analyzer
  - DSLogic Analyzer
  - PulseView

An important factor to consider is that the signal frequency range of the analyzer must be higher than the signals to be analyzed.

5.5.3 USBView

The Universal Serial Bus Viewer (USBView) or *usbview.exe* is a Windows application used to browse all USB controllers and connected USB devices on your PC. USBView works on all versions of Windows.

A USB Device Tree Viewer is also available, which shows more details of devices in the Tree View.

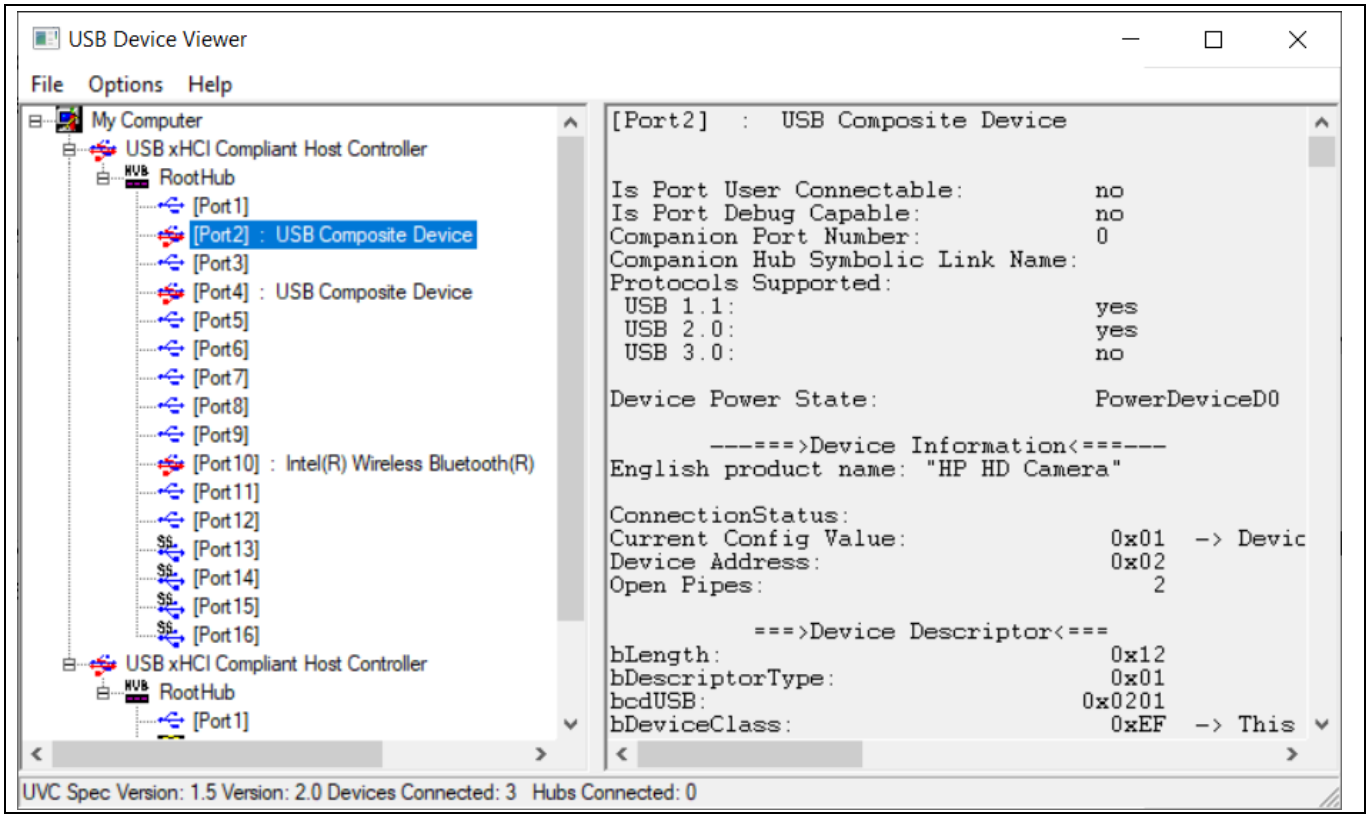

Figure 9 USB Device Viewer - Tree view

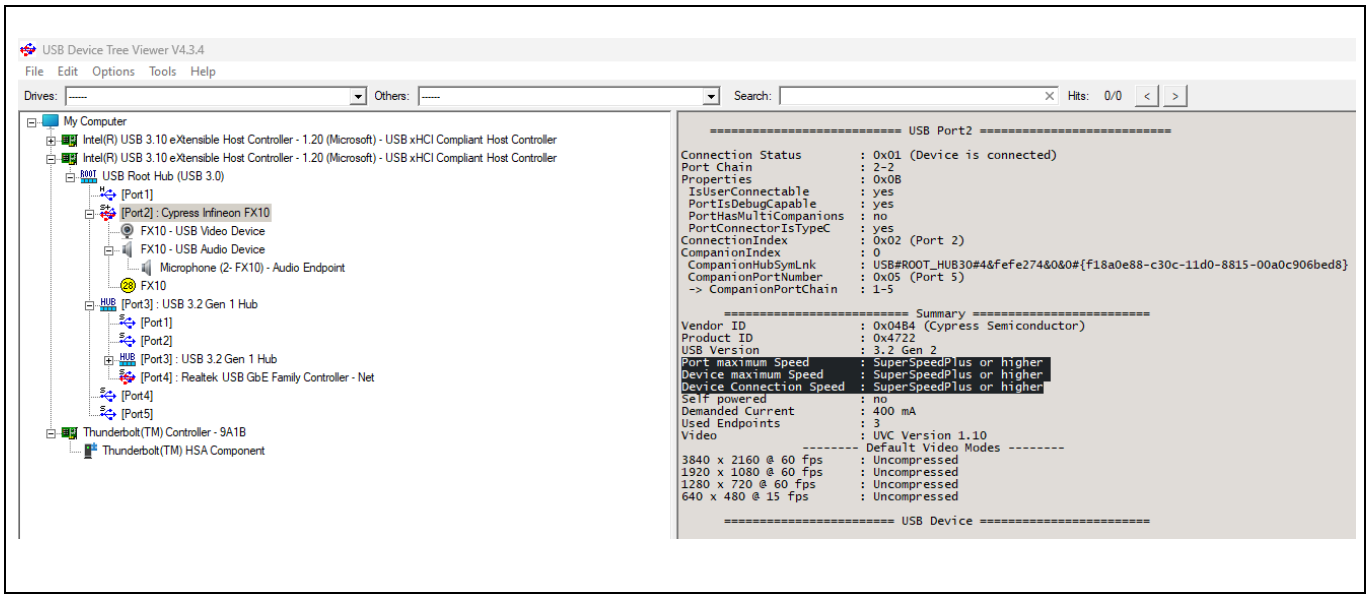

Figure 10 USB Device Tree Viewer - Speeds

5.5.4 UVC host applications

FX20 supports all standard USB Video Class-compatible host applications as follows:

- **Windows:** Windows Camera, MPC-HC, e-CAMView, Webcamoid, VLC
- **Linux:** Cheese player, Webcamoid

- **macOS:** PhotoBooth, Webcamoid

5.5.5 USB3 Vision host applications

FX20 is tested with the following USB3 Vision host applications:

- **Windows:** Pleora eBUS Player, A&B Software (uCAMViewer), Aravis (msys2 lib: aravis-gst)
- **Linux and macOS:** Aravis

5.5.6 Serial terminal (Tera Term, Cool Term, etc.,)

FX20's USB Communication Device Class (CDC) interface can be accessed using the following serial terminal tools:

- **Windows:** Tera Term, PuTTY, Docklight
- **Linux:** screen, CuteCOM, etc.,
- **macOS:** CoolTerm

6 Introduction to the FX20 hardware kit

- **FX20 development kit (DVK):** Provides all interfaces that needs to get started with FX20
- **USB Cable:** Use the USB cable which supports 20 Gbps and dual data lanes

6.1.1 FX20 development kit (KIT_FX20_FMC_001)

The FX20 development kit is designed to help get started with their FX20 design. Use this development kit to evaluate the FX20 code examples. The kit comprises an FX20 baseboard. FX20 baseboard provides clock and voltages for FX20 and high-speed connectors for interfacing external devices. It has one FMC-HPC Mezzanine connector to interface FX20 to any FPGA carrier board and another FMC-HPC carrier connector to route the unused FPGA I/Os to connect to off-the-shelf FMC Mezzanine boards (e.g., high-speed ADC boards). The kit also has a 40-pin GPIO header which can be connected to the GPIO accessory board.

For more details on using this kit, see the FX20 DVK user guide.

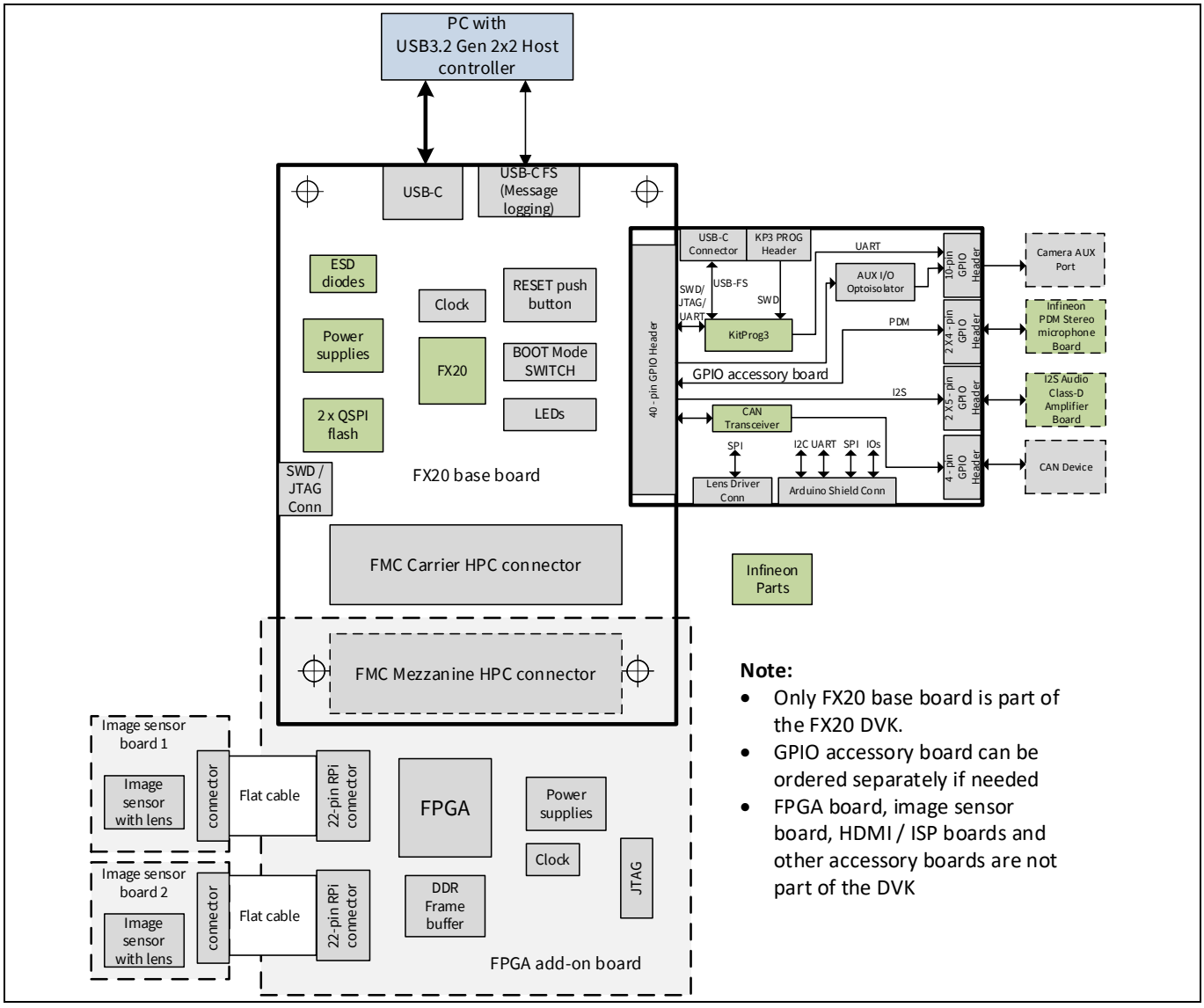

Figure 11 FX20 development kit block diagram

7 Firmware stack

FX20 firmware stack consists of the constituents as shown in Figure 12.

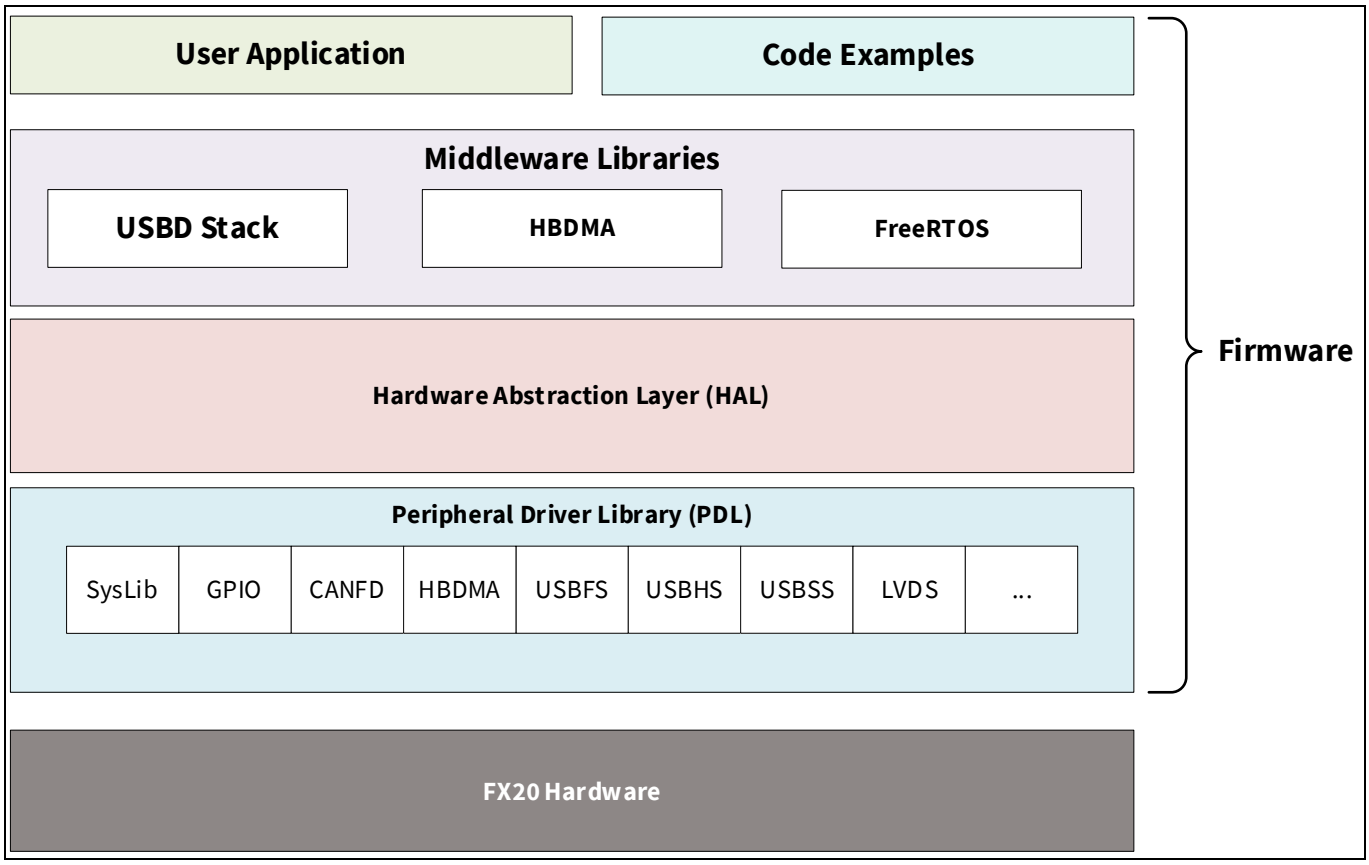

Figure 12 FX20 SDK structure

- Middleware libraries that implement support for FreeRTOS, high bandwidth DMA, and USB protocol stack (USBD) for device controllers
- Hardware abstraction layer (HAL) abstracts out details of the underlying hardware to provide an easy-to-use firmware interface to FX20
- Peripheral driver library provides low-level APIs for each hardware peripheral of FX20

7.1.1 USB bootloader

FX20 has a USB-based bootloader that can be used to program application binaries through its USB interface.

This USB bootloader will be pre-loaded to the internal flash in FX20 at the manufacturing stage. The FX20 bootloader supports device firmware update (DFU) for firmware update.

FX20 bootloader supports USB 2.0 (480 Mbps) operation. If a USB3-based bootloader is required, user can replace this bootloader with custom implementation. User can store the FX20 application firmware in external flash and use the execute-in-place (XIP) feature of FX20's SMIF. See section Quad SPI/serial memory interface (SMIF) for more details.

The bootloader starts running on each power-on reset and checks whether the application firmware in the flash memory is valid. The validity check is performed by computing a SHA256 checksum over the application code region and comparing it with a reference value stored in the metadata region. If the application is not valid or if

a flag indicating that bootloader mode should be entered is set; the bootloader proceeds to enter the firmware update mode. Otherwise, the application binary starts using the information present in the metadata region.

See 002-37666: CYUSB401X, CYUSB308X EZ-USB™ FX10 programming specifications (Contact Infineon support for the document) for more information.

7.1.2 FX20 code examples

Table 3 BSPs for FX20 kits

Sl. no	ModusToolbox™ code example name	Description
1	mtb-example-fx20-hello-world	This code example demonstrates simple USBFS communication by printing a "Hello world" message on a terminal and toggles GPIO using FX20 devices.
2	mtb-example-fx20-usbss-device	This code example demonstrates the implementation of a vendor-specific USB device that allows testing of data transfers using the Bulk, Interrupt, and Isochronous endpoints on USB 2.x and USB 3.2 Gen1/Gen2 interfaces.
3	mtb-example-fx20-uvc-uav	This code example demonstrates the implementation of a UVC 1.1 compliant camera application using the EZ-USB™ FX20 device. An integrated USB Audio Class interface is provided to stream the audio data received from a PDM microphone. This example illustrates the configuration and usage of Sensor Interface Port (SIP) on the FX20 device to implement the Synchronous Slave FIFO IN protocol.

8 Getting started with FX20 application examples

ModusToolbox™ has support for FX20-based application examples. This section provides steps to create examples and generate hex files.

Note: ModusToolbox™ should be version 3.2 or later.

1. Open ModusToolbox™ application and click **New Application**

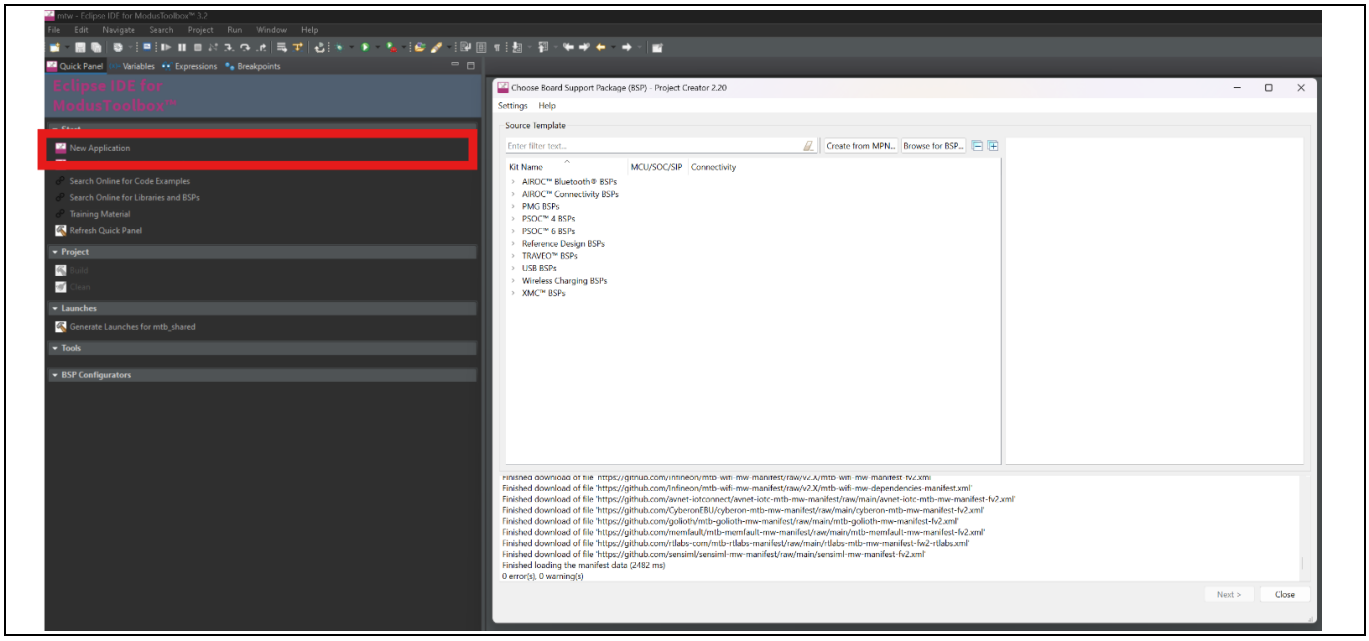

Figure 13 New application in ModusToolbox™

2. Expand USB BSPs in the **Choose Board Support (BSP)** window and select **KIT_FX20_FMC_001**

Getting started with FX20 application examples

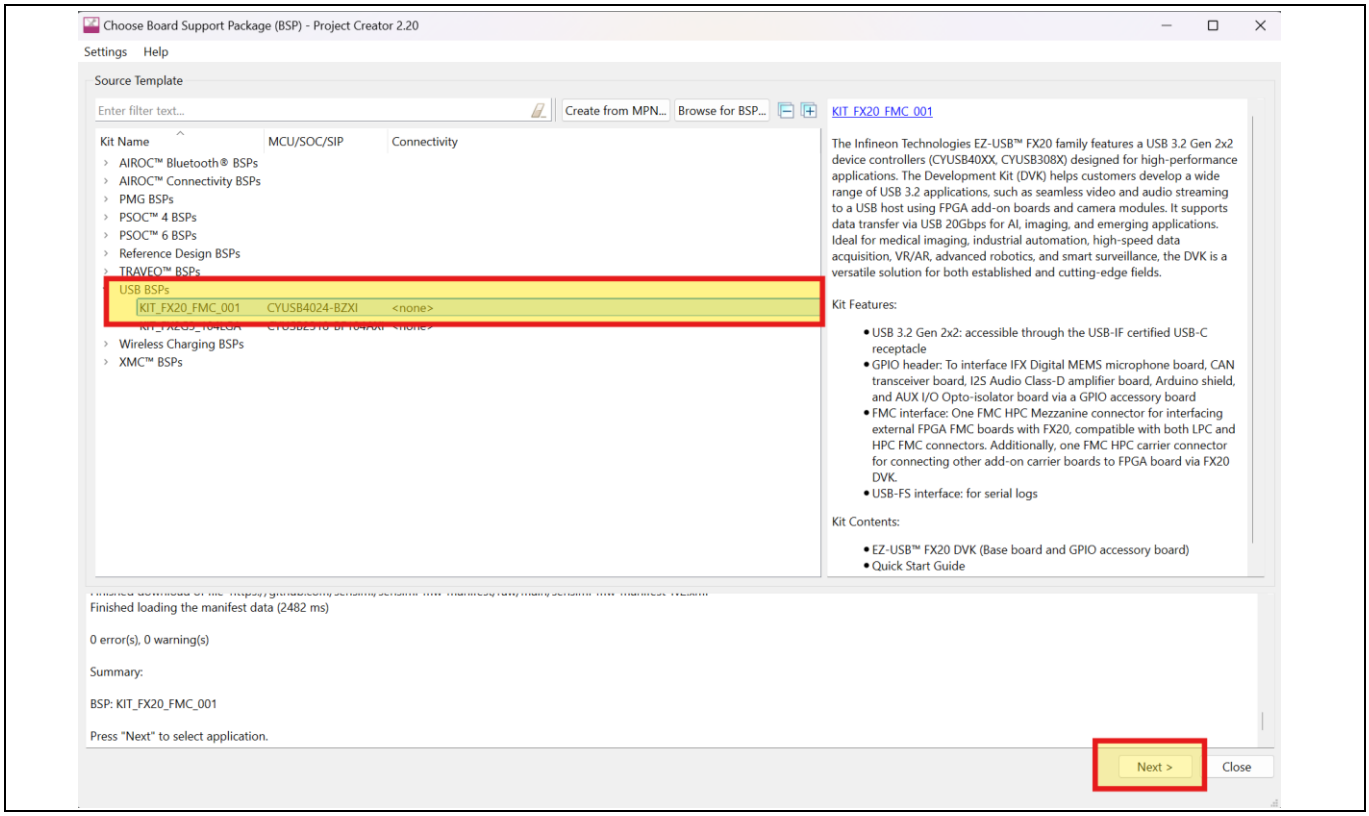

Figure 14 Choose Board Support Package (BSP) – Select KIT_FX20_FMC_001

KIT_FX20_FMC_001 contains an application example which will run on the FX20 hardware kit.

3. Click **Next** to see the template applications developed for FX20

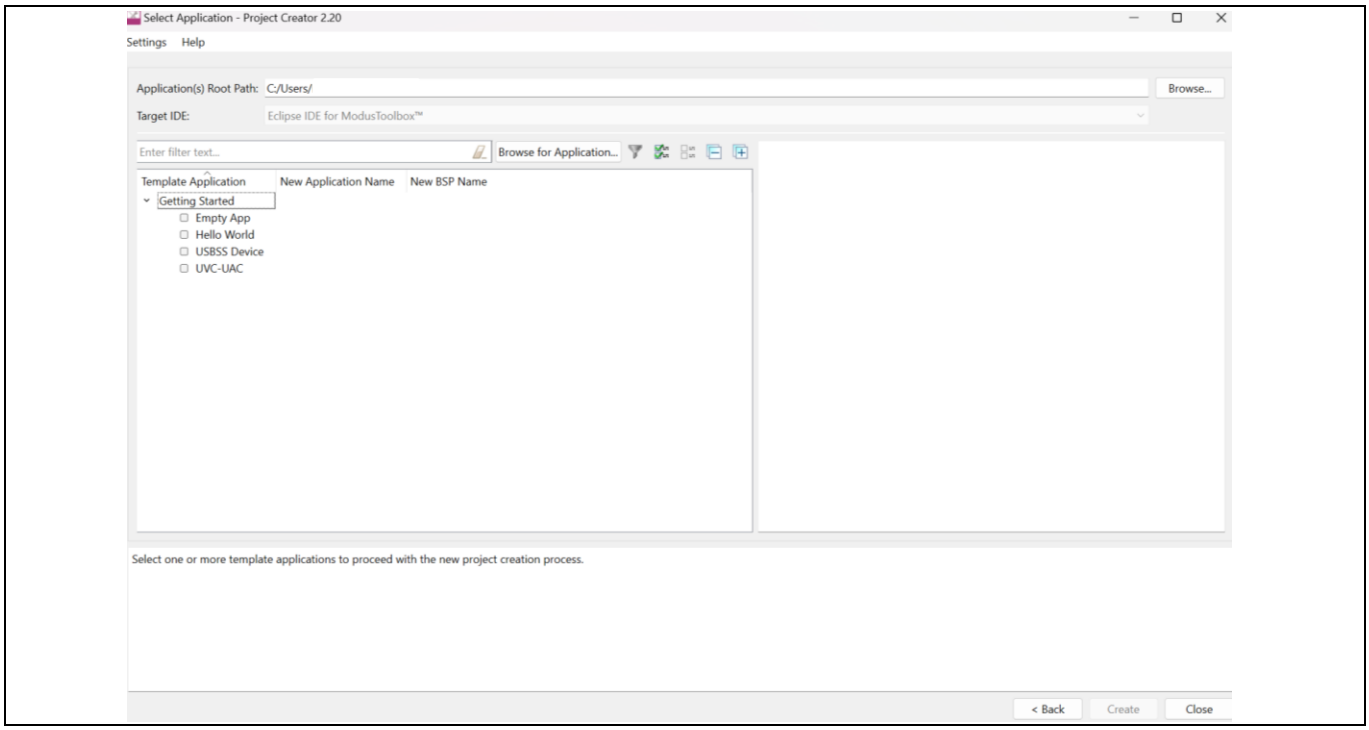

Figure 15 Template Application

8.1 Hello World

This section shows how to build and run a simple firmware example, which prints “Hello World” message over serial host application (Tera-Term).

1. In the **Template Application** section, expand **Getting started** (shown in Figure 15), select **Hello World**, and click **Create** as shown in Figure 16
   Wait for the “Hello World” application to appear in “Project Explorer” of ModusToolbox™.

Note: You can change the default root path if required.

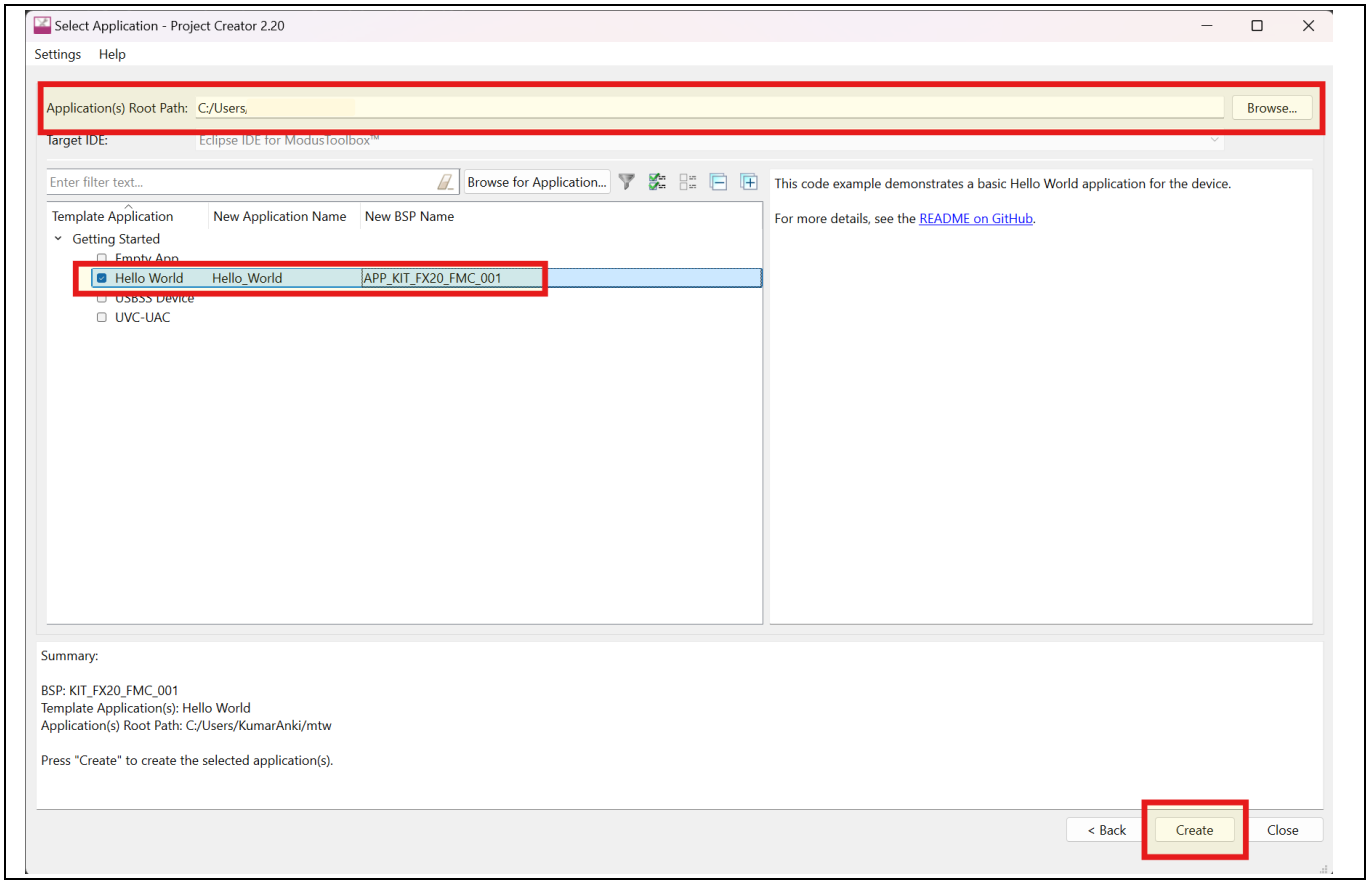

Figure 16 Hello World example

2. When the application is generated, click **Hello World** project and build the application. Example usage details are available in the *README.md* file

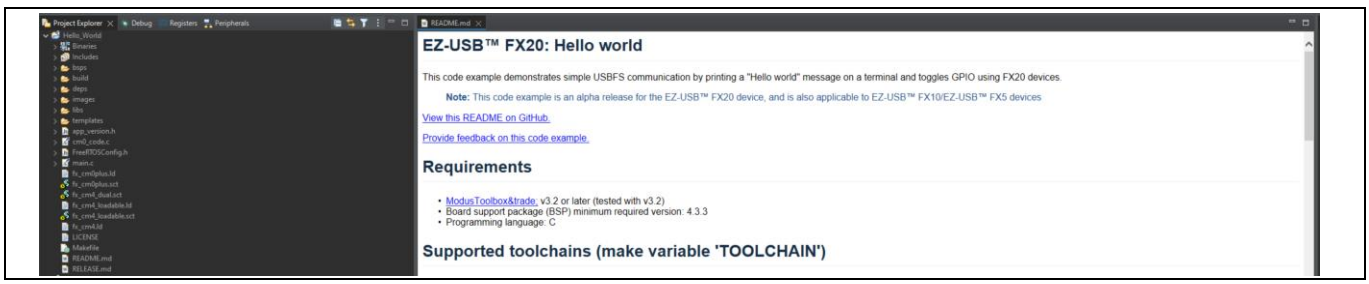

Figure 17 Hello World project README.md

8.2 UVC-UAC

This section explains the steps needed to generate the UVC-UAV example which streams color bar in HOST application using FPGA development kit and FX20 development kit.

1. Select **UVC-UAC** application following the similar steps mentioned in “Hello world” example (Section 8.1) Wait till **UVC-UAC** application appears in **Project Explorer** of ModusToolbox™.

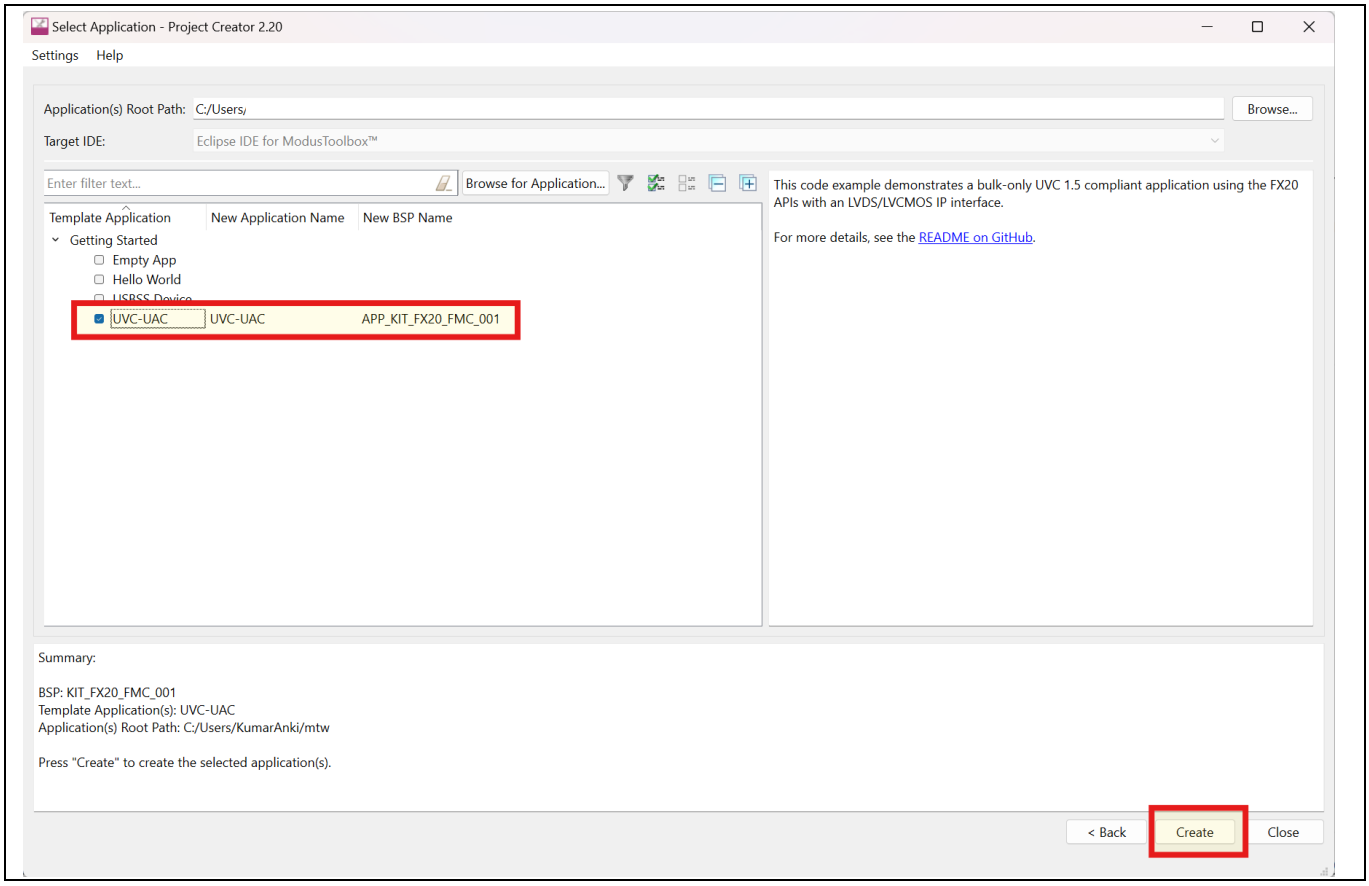

Figure 18 UVC-UAC example

2. When application is generated, click on the **UVC-UAC** project and build the application. Example usage details are available in *README.md* file.

Getting started with FX20 application examples

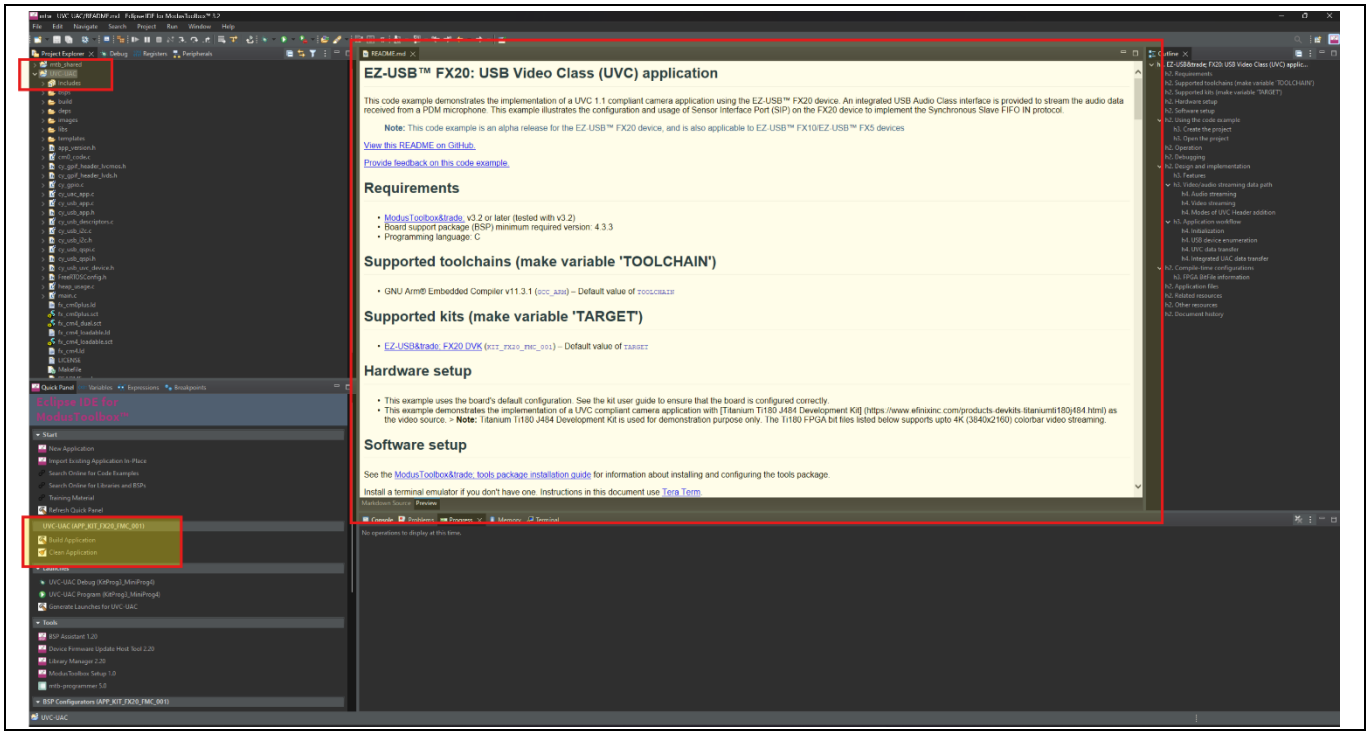

Figure 19 UVC-UAC project README.md

UVC-UAC example can be updated to stream video from the image sensor as well. FX20 is designed to support high-bandwidth data transfers from its LVDS/LVCMOS interface to the USB 20 Gbps interface. This makes it suitable for applications involving high-throughput data inputs such as image sensors and HDMI sources. FX20 can support uncompressed video streams up to 8K at 30 fps or 4K at 120 fps through its LVDS interface. And up to 4K at 60 FPS through its LVCMOS interface.

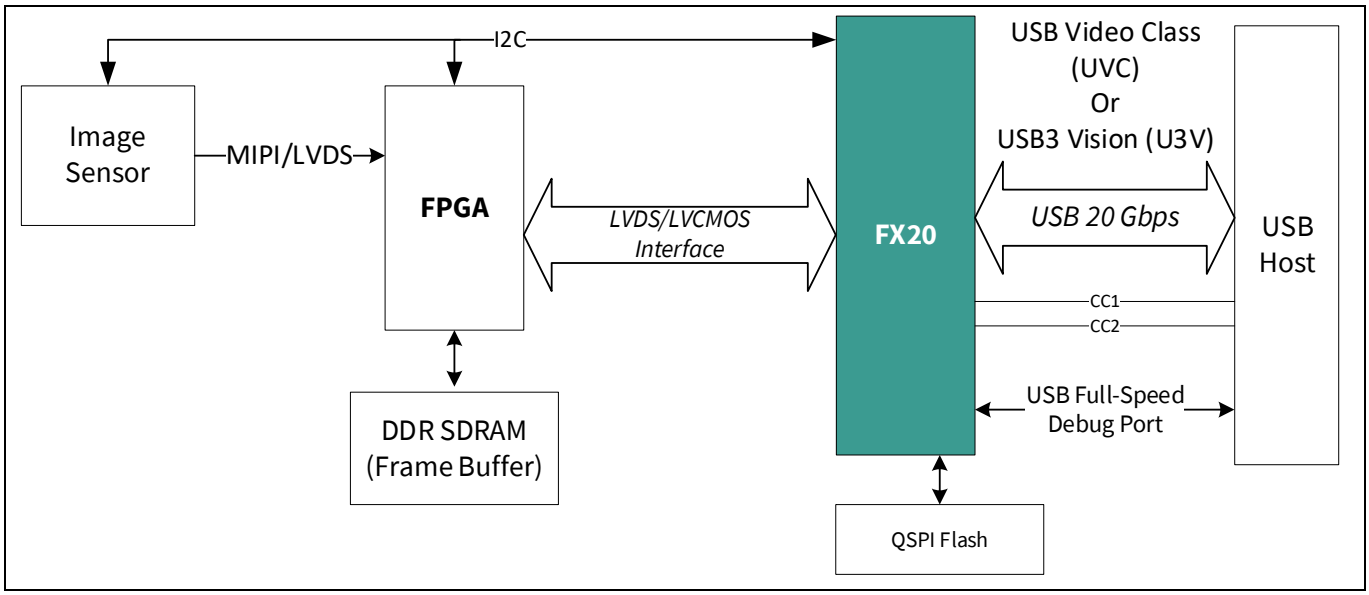

Figure 20 Camera application using FX20

Figure 20 shows a system using FX20 to implement a USB Video Class (UVC)-compatible camera.

9 Migrating the FX3 design to FX5 or FX20

9.1 Migrating FX3-based hardware design to FX20

The following application notes describe the hardware design guidelines for FX20 and FX3.

- AN70707 - EZ-USB™ FX3 hardware design guidelines and schematic checklist
- AN238422 - EZ-USB™ FX5/FX20 hardware design guidelines and schematic checklist

9.1.1 Power supply and I/O voltage comparison

Table 4 Power supply and I/O voltage comparison

Description	Parameter		Typical		Unit
	FX3	FX20	FX3	FX20	
Core supply voltage	VDD/ AVDD	VDDD	1.8	1.8/3.3	V
IO supply voltage	VIO	VDDIO	1.8	1.8	V
USB3 supply	$U3TX_{VDDQ}$, $U3RX_{VDDQ}$	USB3V18	1.8	1.8	V

9.1.2 Crystal/clock

FX20 supports only XTAL input as compared to FX3. Table 5 shows the clocking options supported by the FX3 and FX20.

Table 5 Clock options

	FX3	FX20
Crystal	19.2 MHz	24 MHz
External clock	19.2, 26, 38.4, and 52 MHz	24 MHz

9.2 Migrating FX3-based firmware application to FX20

An existing FX3 application can be migrated based on any of the following code examples to FX20 by choosing the appropriate code example as a starting point in ModusToolbox™.

Table 6 FX3 code examples and corresponding FX20 code example

FX3 code example	FX20 code example	Description
AN75779 UVC Firmware	mtb-example-fx20-uvc-uac	Demonstrates USB Video Class (UVC) application with FX3. The FX20 code example demonstrates a USB composite device with both UVC and USB Audio Class (UAC) interfaces.

Note: The GPIF II state machines created for FX3 are not compatible with FX20.

Migrating the FX3 design to FX5 or FX20

FX20 supports the following interfaces on the LVDS / LVCMOS interface:

Table 7 FX20 supported interfaces on LVDS/LVCMOS

Mode	Supported interface
LVDS	Normal streamer with metadata insertion
LVCMOS	2-bit Slave FIFO
	5-bit Slave FIFO

References

For a complete and updated list of EZ-USB™ FX20 code examples, visit our GitHub repository. For more EZ-USB™ FX20 related documents, visit the EZ-USB™ FX20 webpage.

Application note

[1] Infineon Technologies AG: *AN237841 – Getting started with EZ-USB™ FX20/FX10/FX5N/FX5*

[2] Infineon Technologies AG: *AN75779 - How to implement an image sensor interface using EZ-USB™ FX3 in a USB Video Class (UVC) framework;* Available online

[3] Infineon Technologies AG: *AN238689 - Implementing a USB video and audio composite device using EZ-USB™ FX20/FX10/FX5N/FX5 controller;* Available online

[4] Infineon Technologies AG: *AN241535 - Designing LVDS transmitter in FPGA to interface with EZ-USB™ FX20/FX10/FX5/FX5N controller;* Available online

Datasheet

[5] Infineon Technologies AG: *002-33684: CYUSB402x EZ-USB™ FX20 USB 20 Gbps peripheral controller;* Available online

[6] Infineon Technologies AG: *002-40837: CYUSB401x EZ-USB™ FX10 USB 10 Gbps peripheral controller;* Available online

[7] Infineon Technologies AG: *002-41019: CYUSB328x EZ-USB™ FX5N USB 10 Gbps peripheral controller;* Available online

[8] Infineon Technologies AG: *002-40850: CYUSB308x EZ-USB™ FX5 USB 5 Gbps peripheral controller;* Available online

Reference manual

[9] Infineon Technologies AG: *002-36077: EZ-USB™ FX20 USB 3.2 Gen 2x2 device controller architecture reference manual;* Available online

Programming specifications

[10] Infineon Technologies AG: *002-37666: CYUSB401x, CYUSB308x EZ-USB™ FX20 programming specifications*

Development kits

[11] Infineon Technologies AG: *KIT_FX20_FMC_001: EZ-USB™ FX20 Development Kit;* Available online

[12] Infineon Technologies AG: *KIT_FX10_FMC_001: EZ-USB™ FX10 Development Kit*

[13] Infineon Technologies AG: *KIT_FX5N_FMC_001: EZ-USB™ FX5N Development Kit*

[14] Infineon Technologies AG: *KIT_FX5_FMC_001: EZ-USB™ FX5 Development Kit*

[15] Infineon Technologies AG: *DEMO_FX20_MIPI_001: EZ-USB™ FX20 MIPI Camera Demo Kit*

Tools

[16] Infineon Technologies AG: *ModusToolbox™ Tools Package*; Available online

References

[17] Infineon Technologies AG: *EZ-USB™ FX Control Center*; Available online

Product webpages

[18] Infineon Technologies AG: *EZ-USB™ FX20 webpage*; Available online

[19] Infineon Technologies AG: *EZ-USB™ FX10/FX5N webpage*; Available online

[20] Infineon Technologies AG: *EZ-USB™ FX5 webpage*; Available online

Glossary

AES

Advanced Encryption Standard (AES)

BGA

ball grid array (BGA)

CRC

cyclic redundancy check (CRC)

CDC

communication device class (CDC)

DDR

Double Data Rate (DDR)

HBWDMA

high bandwidth direct memory access (HBWDMA)

HBWSS

high bandwidth data subsystem (HBWSS)

HID

human interface device (HID)

LVCMOS

low voltage CMOS (LVCMOS)

LVDS

low voltage differential signalling (LVDS)

Glossary

PTM

precision time measurement (PTM)

SCB

Serial Communication Block (SCB)

SDR

Single Data Rate (SDR)

SHA

Secure Hash Algorithm (SHA)

UVC

USB Video Class

U3V

USB3 Vision

Revision history

Document revision	Date	Description of changes
**	2025-03-11	Initial release
*A	2025-06-06	Updated reference links

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2025-06-06

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Email: erratum@infineon.com

Document reference

002-37841 Rev. *A

Important notice

The information contained in this application note is given as a hint for the implementation of the product only and shall in no event be regarded as a description or warranty of a certain functionality, condition or quality of the product. Before implementation of the product, the recipient of this application note must verify any function and other technical information given herein in the real application. Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this application note.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

Warnings

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.