VISION SDK OVERVIEW
S32V234 SDK

AMF-AUT-T2323

BRYAN THOMAS

FIELD APPLICATION ENGINEER

SECURE CONNECTIONS FOR A SMARTER WORLD

PUBLIC

Outline

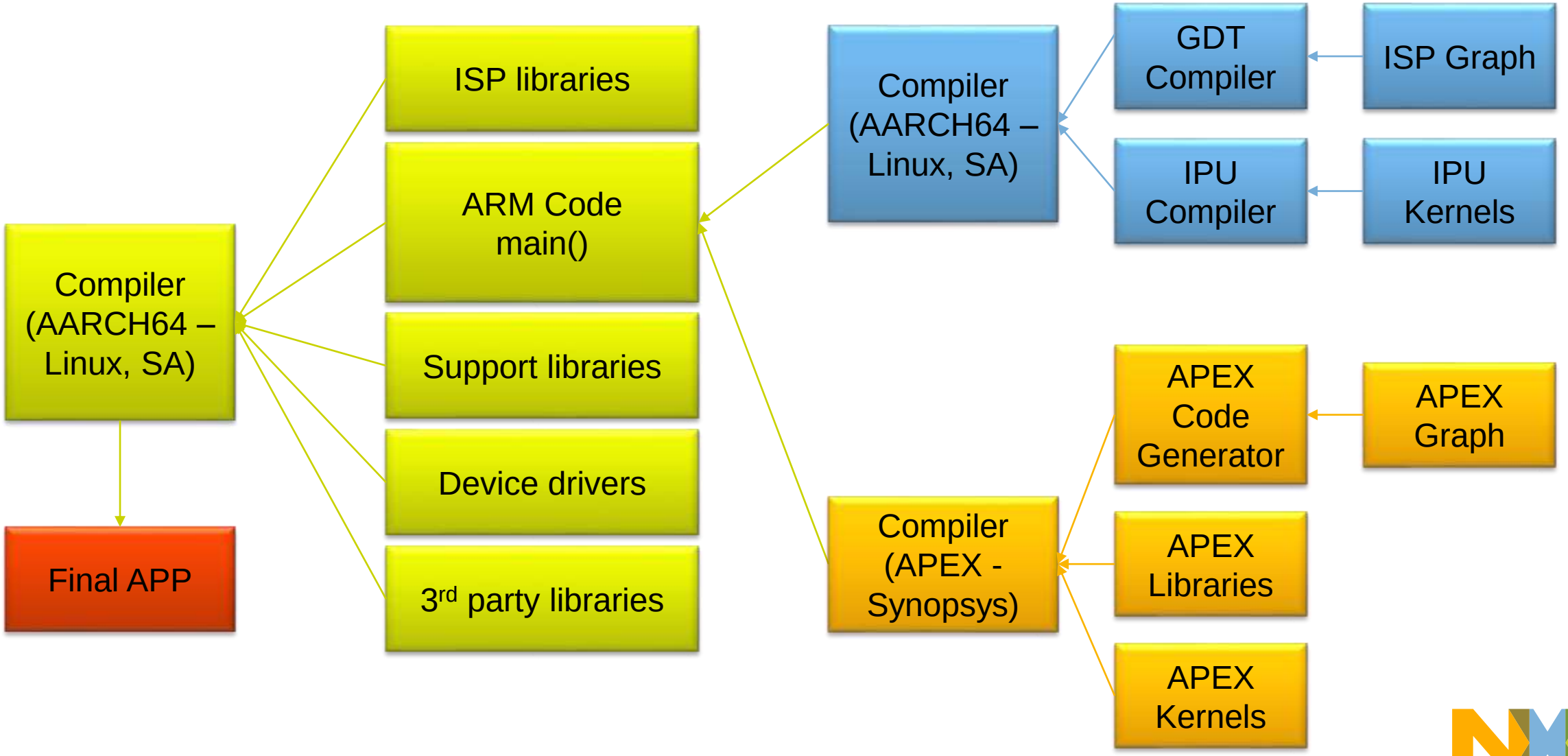
- Part 1
  1. VSDK overview and application component mapping
  2. 3rd party libraries
  3. NXP Device drivers and libraries
  4. ISP Related resources
  5. APEX Related resources
  6. Compilers available
  7. Build system

- Part 2
  1. Documentation
  2. Demos
  3. OS related content
  4. Running the demos
  5. Debugging

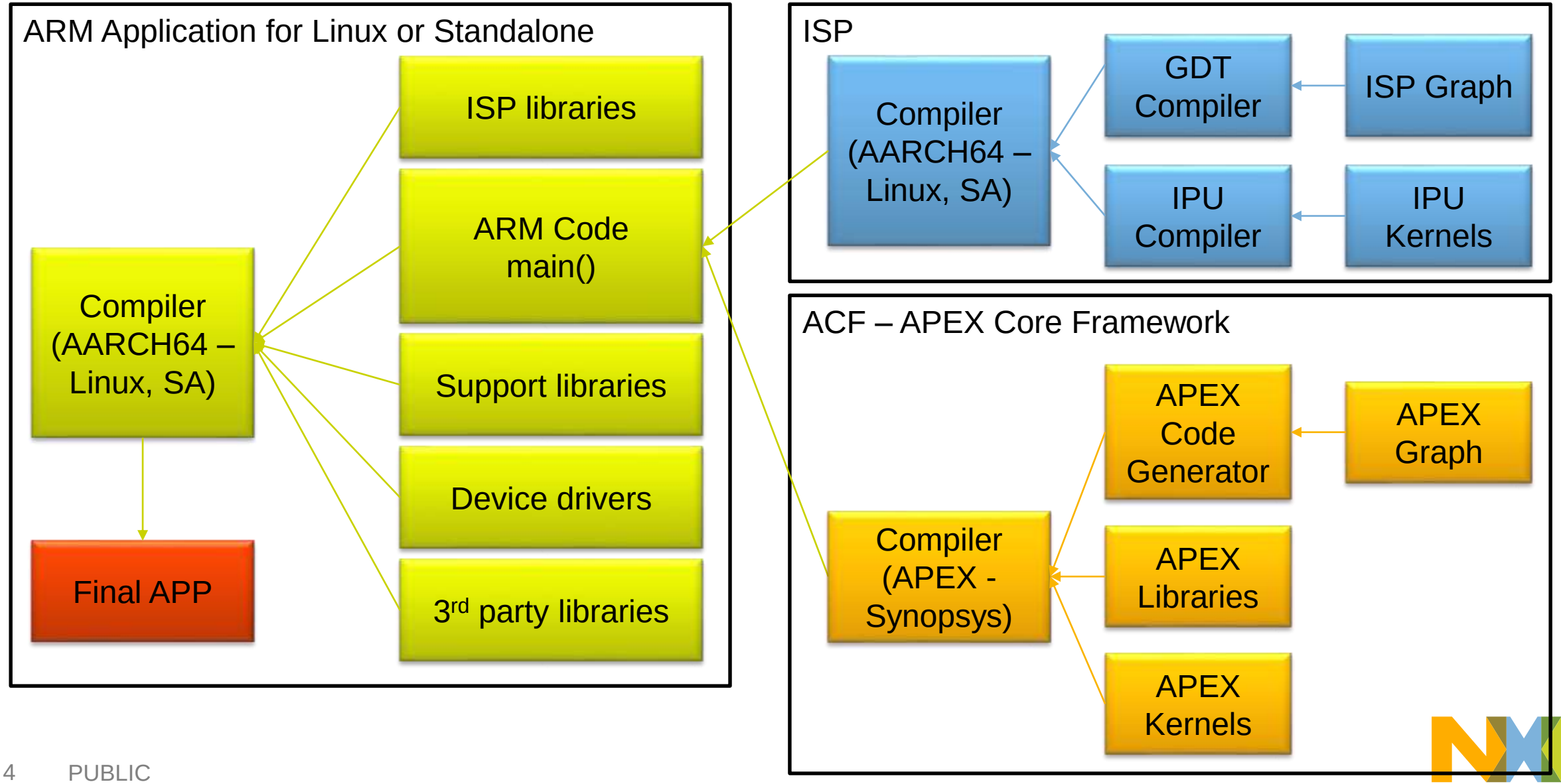
Part 1

1. VSDK overview and application component mapping
2. 3rd party libraries
3. NXP Device drivers and libraries
4. ISP Related resources
5. APEX Related resources
6. Compilers available
7. Build system

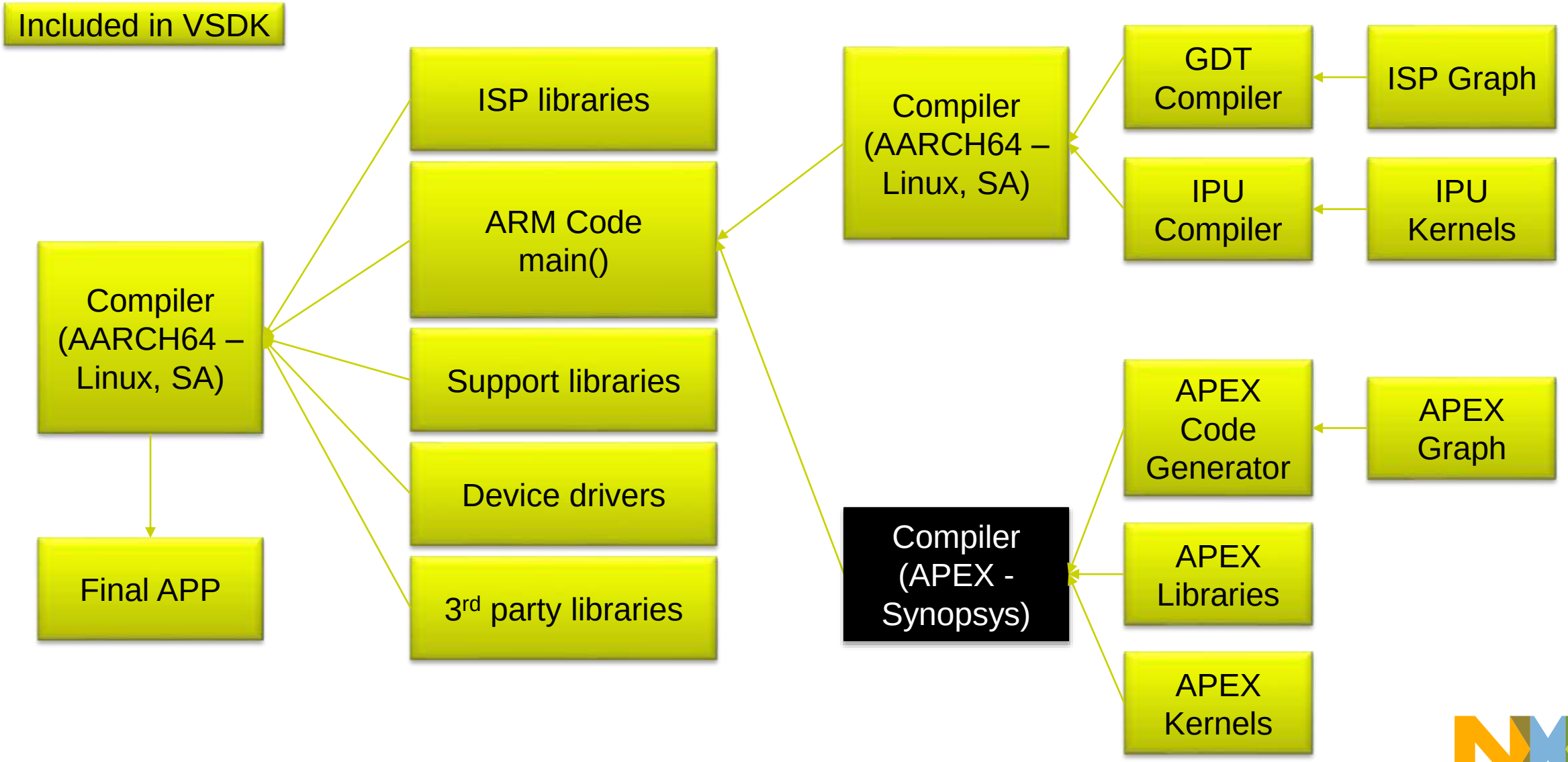
1.1 Application component overview



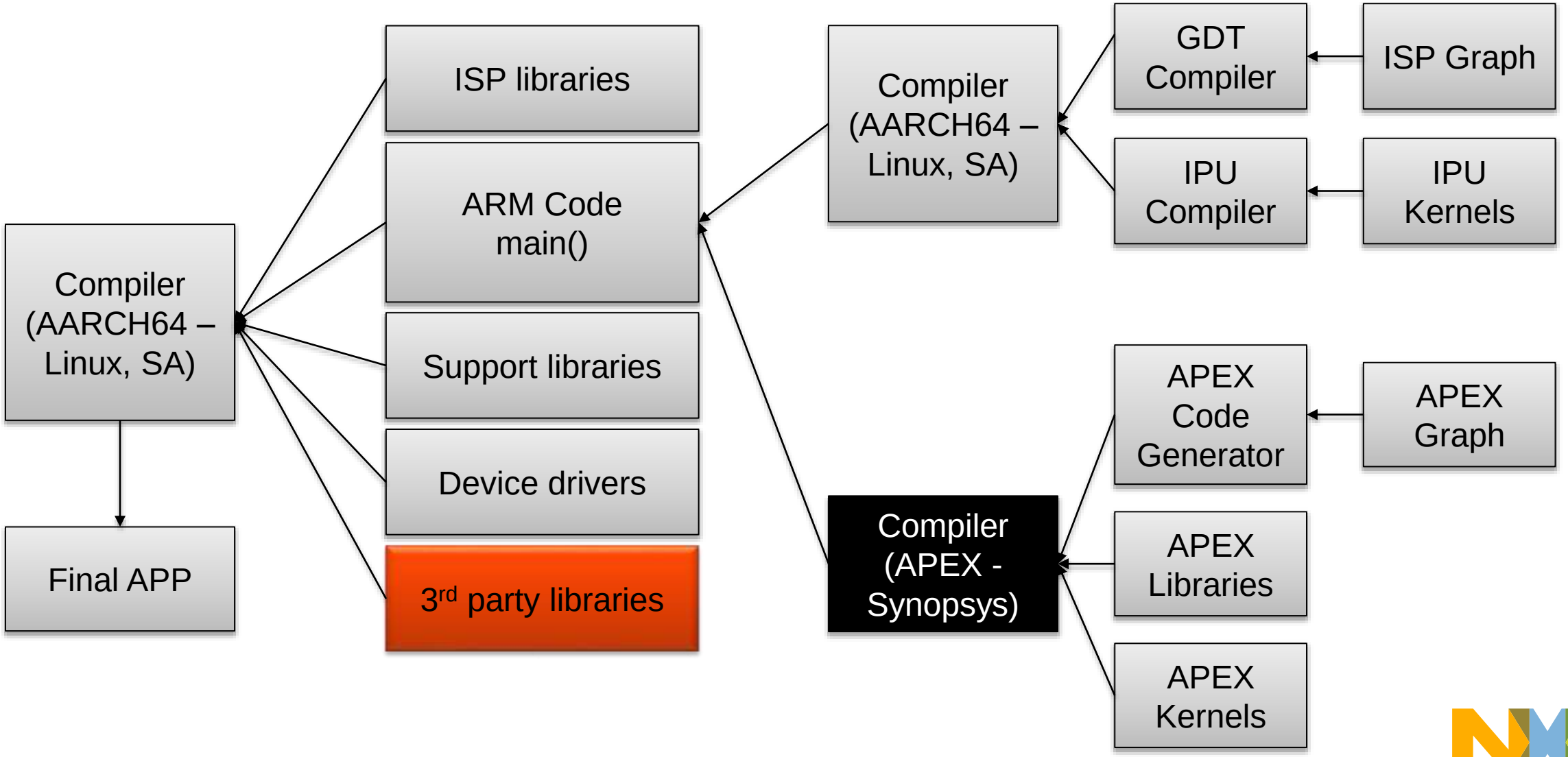
1.1 Application component overview



1.1 Application component overview – VSDK Content

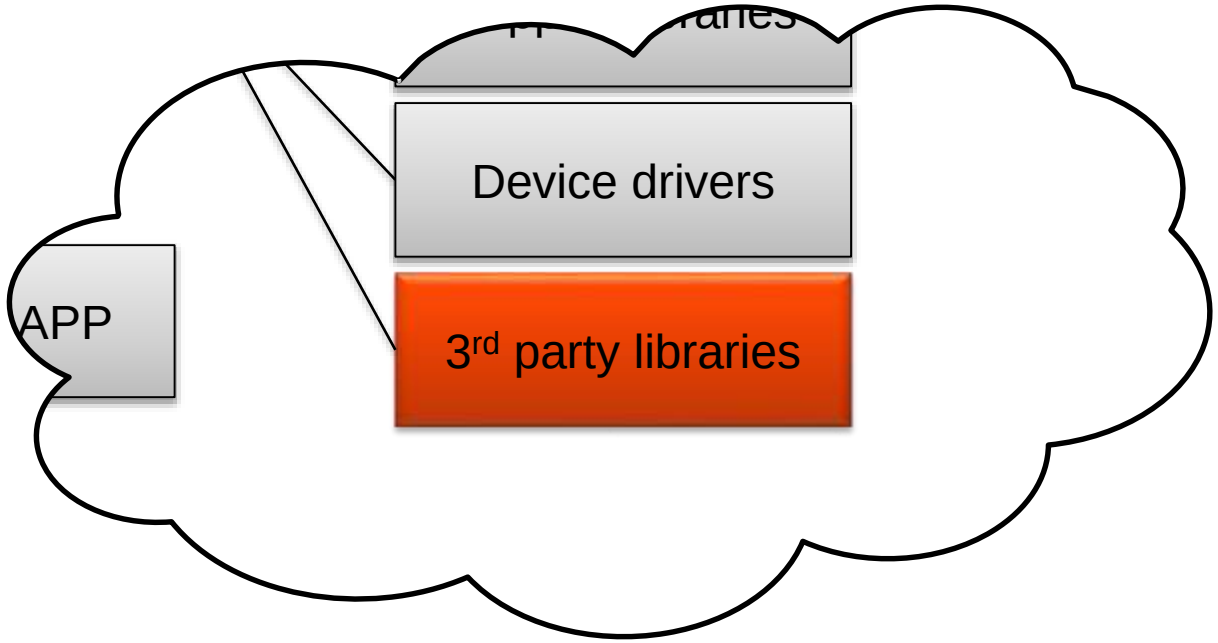


1.2 3rd Party Libraries



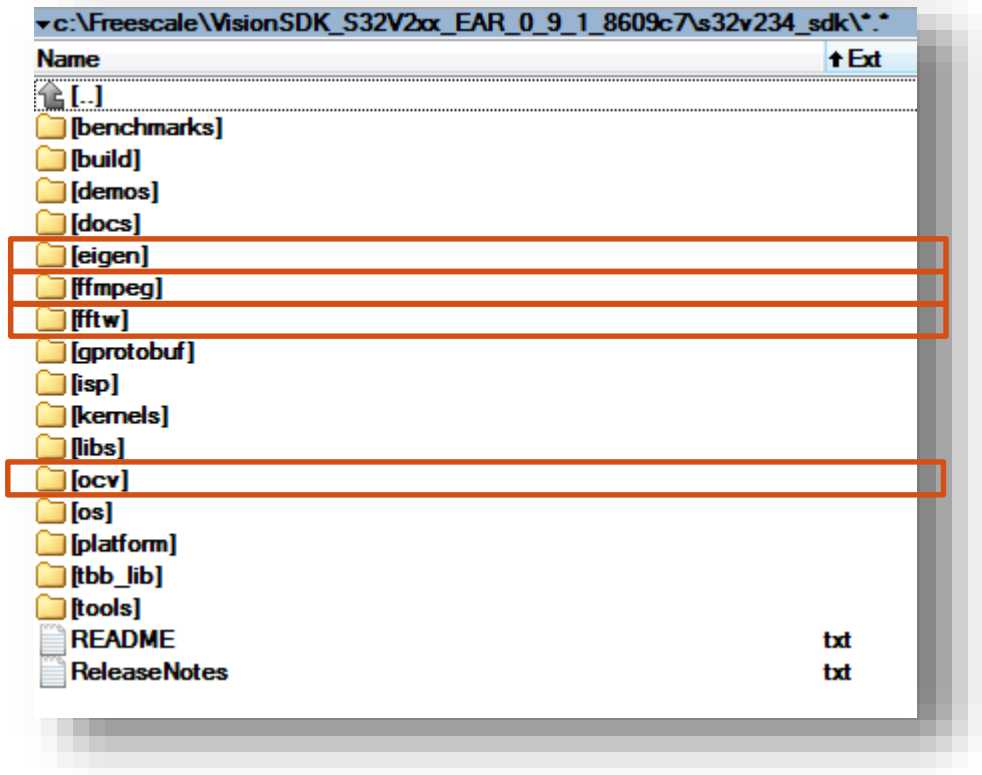
1.2 3rd Party Libraries

- Provided in the VSDK
  - OpenCV 2.4.10
    - Built for Linux + Standalone
    - Scripts available for rebuild
  - FFMPEG 2.2.1
    - Built for Linux + Standalone
    - Scripts available for rebuild
  - FFTW 3.3.4
    - Script available for Standalone
  - Eigen
    - Headers available to be included to the application

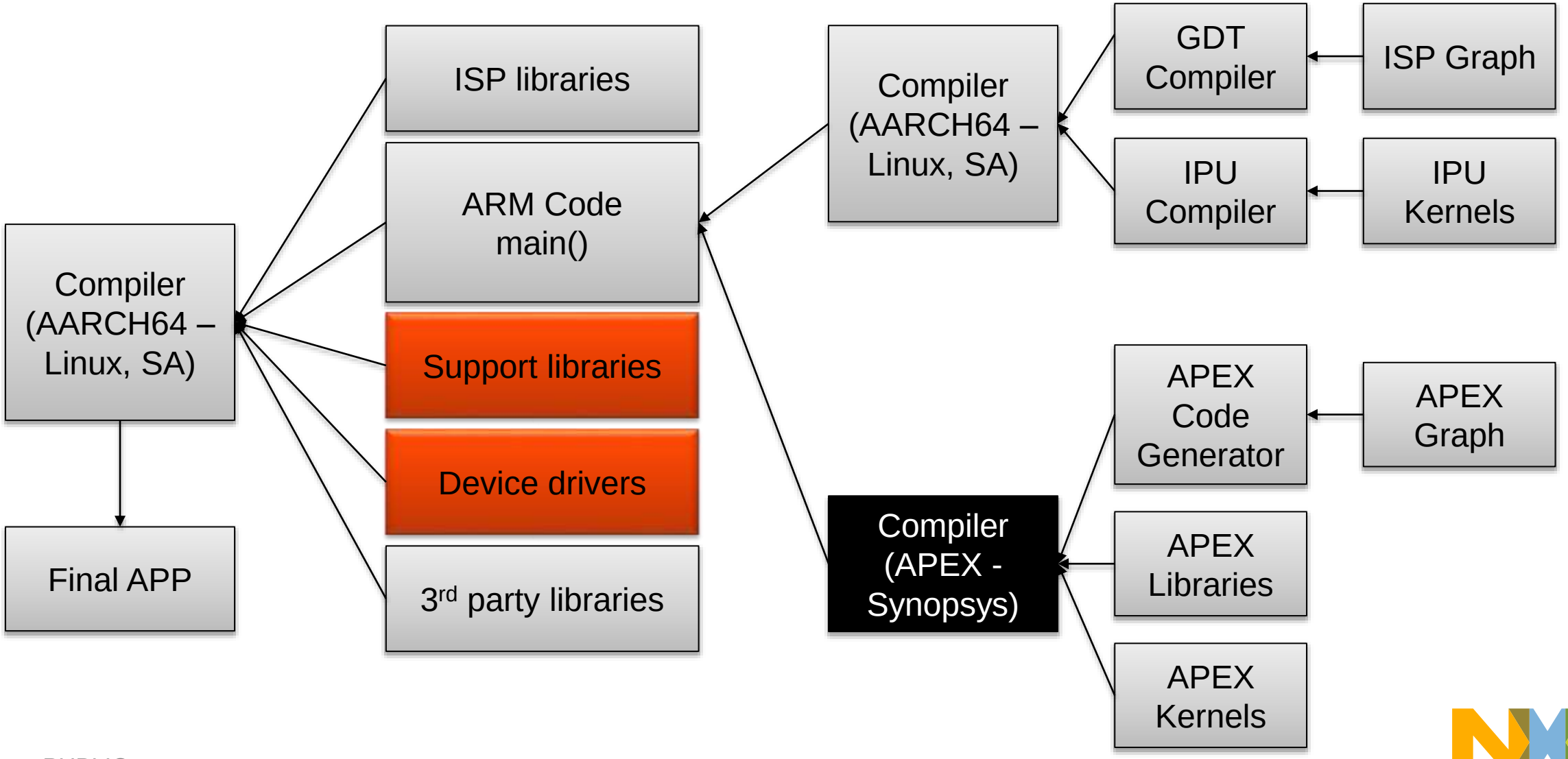


1.2 3rd party libraries

- Provided in the VSDK
  - OpenCV 2.4.10
    - Built for Linux + Standalone
    - Scripts available for rebuild
  - FFMPEG 2.2.1
    - Built for Linux + Standalone
    - Scripts available for rebuild
  - FFTW 3.3.4
    - Script available for Standalone
  - Eigen
    - Headers available to be included to the application



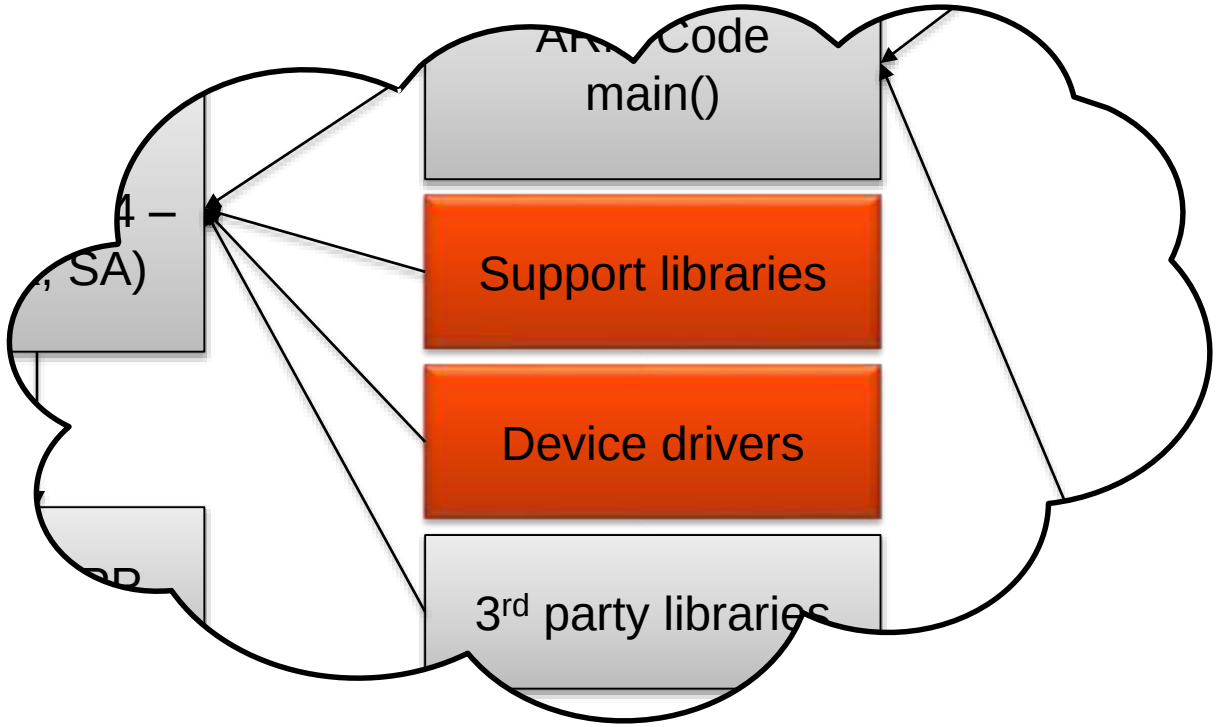
1.3 NXP Device drivers and libraries

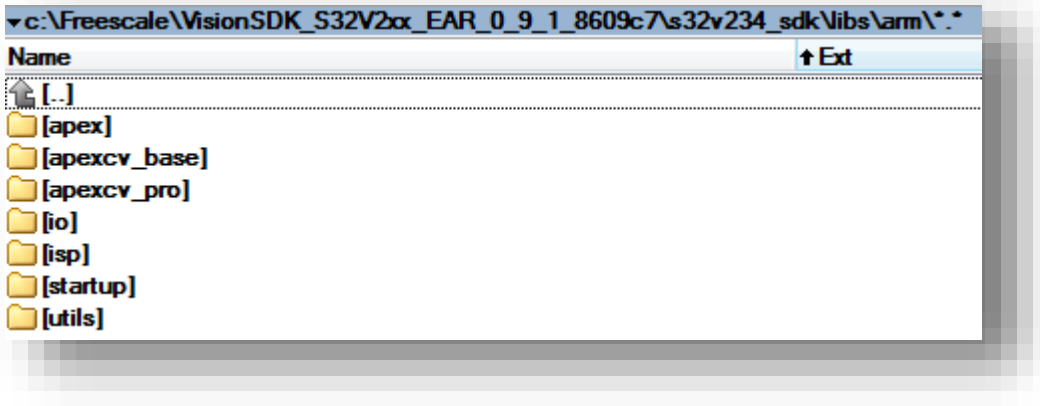


1.3 NXP Device drivers and libraries

- Provided in the VSDK with source code
  - APEX related libs
  - APEXCV Base
  - APEXCV Pro

    (pre-built only in ApexCV Pro release)
  - I/O
  - ISP related libs
  - Startup (for Standalone)
  - Utils (Auxiliary utility libs)





1.3 NXP Device drivers and libraries - APEX related libs

- **Apex Core Framework library**
  - C++ Wrapper for APEX Graphs
  - Inherited class to ACF Process is used in every built ACF Graph (separate training)
- **APEX Driver**
  - kernel module + user space lib in case of Linux
  - library in case of Standalone application
- **ICP**
  - Image Cognition Datatypes to be used with ACF
- **OpenCL Driver for APEX**
  - kernel module + user space lib in case of Linux
  - library in case of Standalone application

Provided in the VSDK with source code

APEX related libs

APEXCV Base

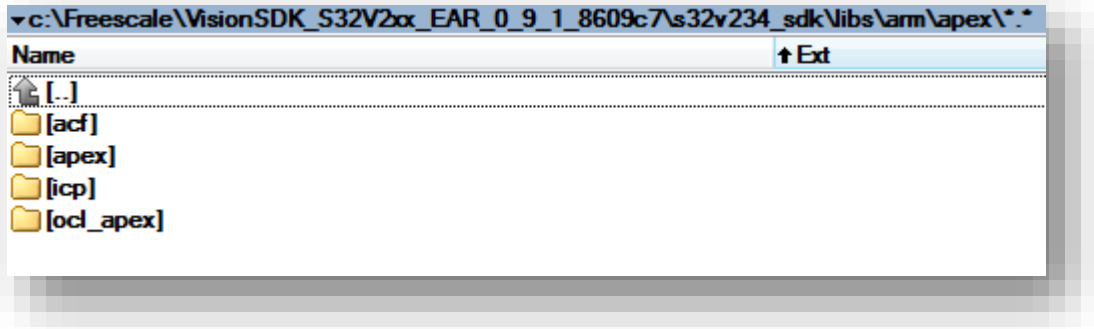
APEXCV Pro

I/O

ISP related libs

Startup (for Standalone)

Utils (Auxiliary utility libs)



1.3 NXP Device drivers and libraries – APEXCV Base

- **ApexCV Core**
  - Common headers and data types for ApexCV
- **Arithmetic library**
- **Color Conversion library**
- **Histogram library**
- **Image Filtering library**
- **Integral Image library**
- **Interpolation library**

Provided in the VSDK with source code

APEX related libs

APEXCV Base

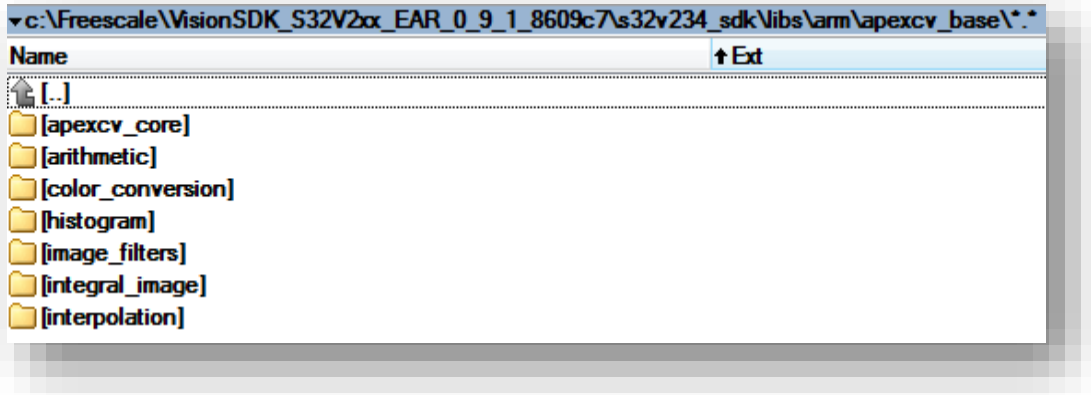
APEXCV Pro

I/O

ISP related libs

Startup (for Standalone)

Utils (Auxiliary utility libs)



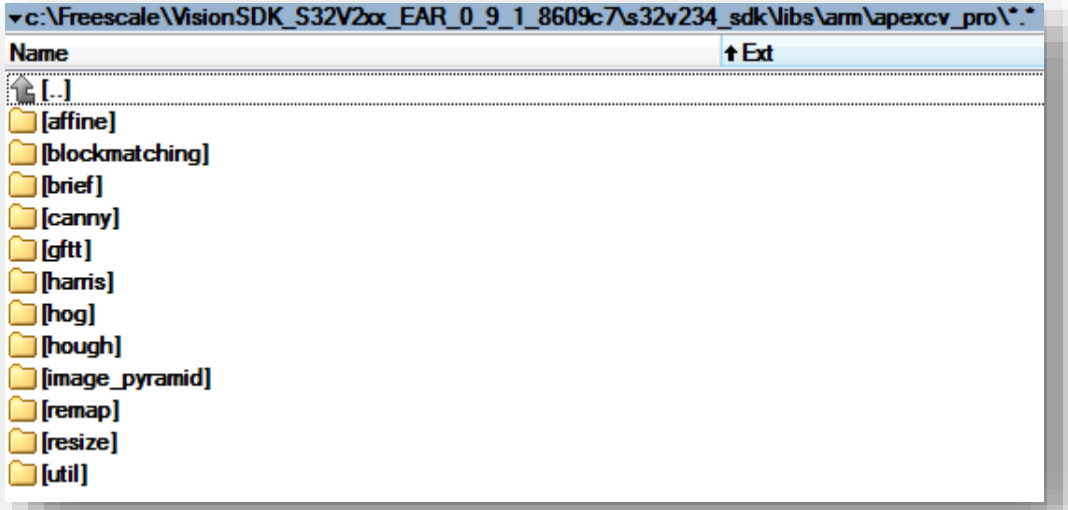
1.3 NXP Device drivers and libraries – APEXCV Pro

- **Provided as prebuilt libraries and only in ApexCV Pro package**

- **Affine Transformations**
- **BRIEF**
- **Canny detector**
- **Good Features to Track**
- **Harris detector**
- **HOG**
- **Hough Transformation**
- **Image Pyramid**
- **Image Remapping**
- **Image Resize**
- **Util**
  - Utility library common to every module

Provided in the VSDK with source code

- APEX related libs
- APEXCV Base
- **APEXCV Pro**
- I/O
- ISP related libs
- Startup (for Standalone)
- Utils (Auxiliary utility libs)

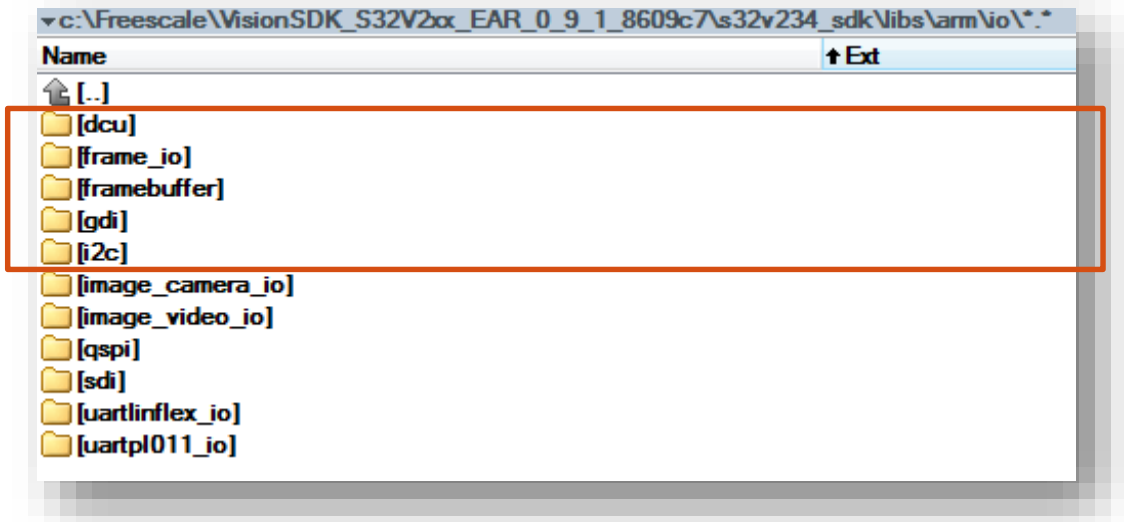


1.3 NXP Device drivers and libraries – I/O Part 1

- **DCU Driver**
  - Display Controller Unit Driver
  - Used for image output
- **Frame I/O Wrapper**
  - Wrapper for simple image output to several devices (DCU, file, memory etc.)
- **Framebuffer driver for Linux**
  - kernel module + user space lib
- **GDI**
  - Graphical Data Interface (**deprecated**)
- **I2C Driver**
  - I2C Driver for Standalone applications

Provided in the VSDK with source code

- APEX related libs
- APEXCV Base
- APEXCV Pro
- **I/O**
- ISP related libs
- Startup (for Standalone)
- Utils (Auxiliary utility libs)

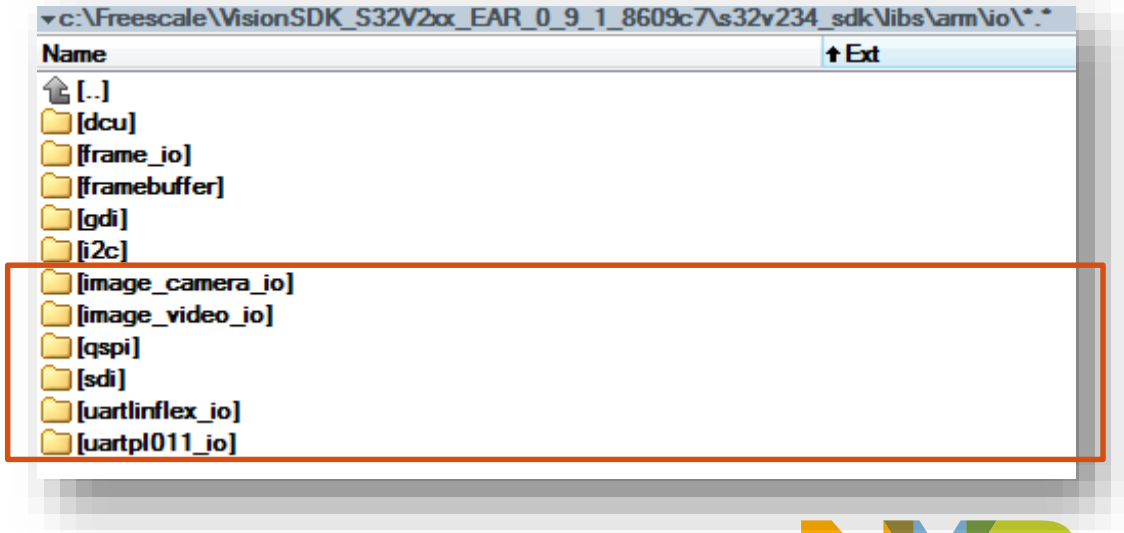


1.3 NXP Device drivers and libraries – I/O Part 2

- **Image Camera I/O**
  - Old library for input from camera (**deprecated**)
  - Frame I/O used instead
- **Image Video I/O**
  - Old library for I/O from/to video file (**deprecated**)
  - Frame I/O used instead
- **QSPI**
  - QSPI Driver for standalone applications
- **SDI**
  - Sensor Device Interface
  - Will replace Frame I/O for camera in future releases
- **UART Linflex Driver**
  - UART Driver for standalone applications
- **UART pl011 Driver**
  - UART Driver for ZYNQ standalone applications (**deprecated**)

Provided in the VSDK with source code

- APEX related libs
- APEXCV Base
- APEXCV Pro
- **I/O**
- ISP related libs
- Startup (for Standalone)
- Utils (Auxiliary utility libs)

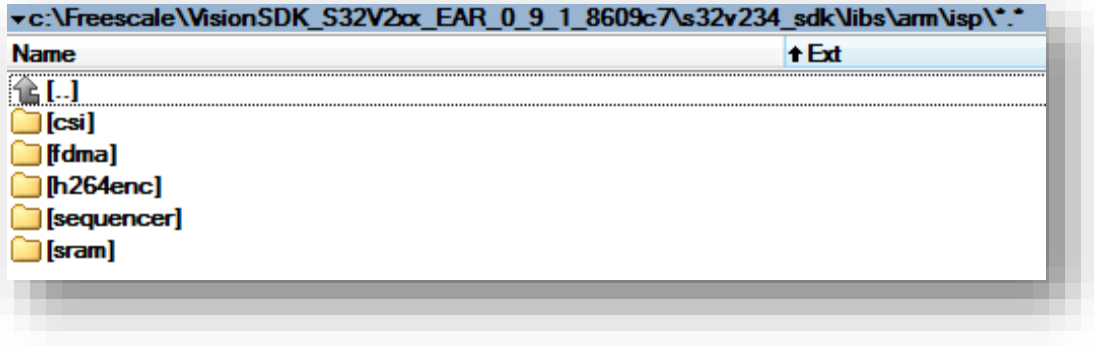


1.3 NXP Device drivers and libraries – ISP related libs

- **MIPI CSI Driver**
  - kernel module + user space lib in case of Linux
  - library in case of Standalone application
- **FDMA Driver**
  - kernel module + user space lib in case of Linux
  - library in case of Standalone application
- **H264 Encoder Driver**
- **Sequencer Driver**
  - kernel module + user space lib in case of Linux
  - library in case of Standalone application
- **SRAM Driver**
  - kernel module + user space lib in case of Linux
  - library in case of Standalone application

Provided in the VSDK with source code

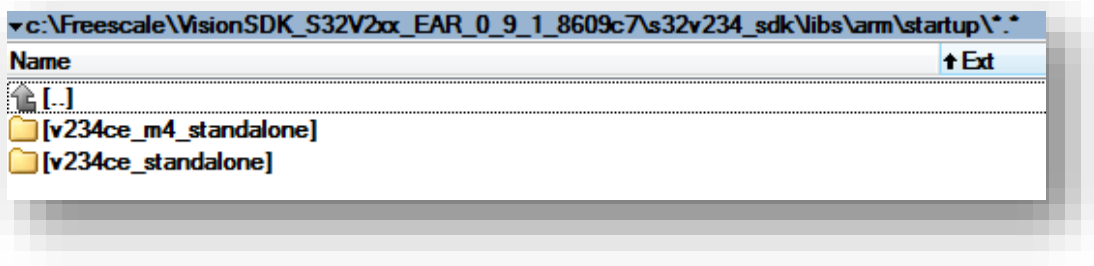
APEX related libs
APEXCV Base
APEXCV Pro
I/O
ISP related libs
Startup (for Standalone)
Utils (Auxiliary utility libs)



1.3 NXP Device drivers and libraries – Startup libs

- **M4 Startup**
  - Pre-compiled
  - Can be recompiled with ARM compiler (not supported)
  - Contains the code for M4 startup
  - Used automatically when app is loaded via debugger
- **A53 Startup**
  - Contains the code for A53 entry on standalone

Provided in the VSDK with source code
- APEX related libs
- APEXCV Base
- APEXCV Pro
- I/O
- ISP related libs
- **Startup (for Standalone)**
- Utils (Auxiliary utility libs)

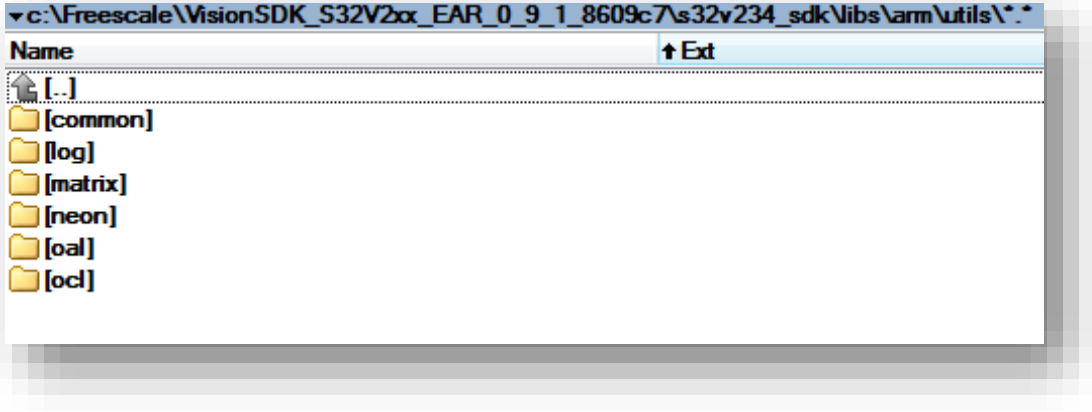


1.3 NXP Device drivers and libraries – Utility libraries

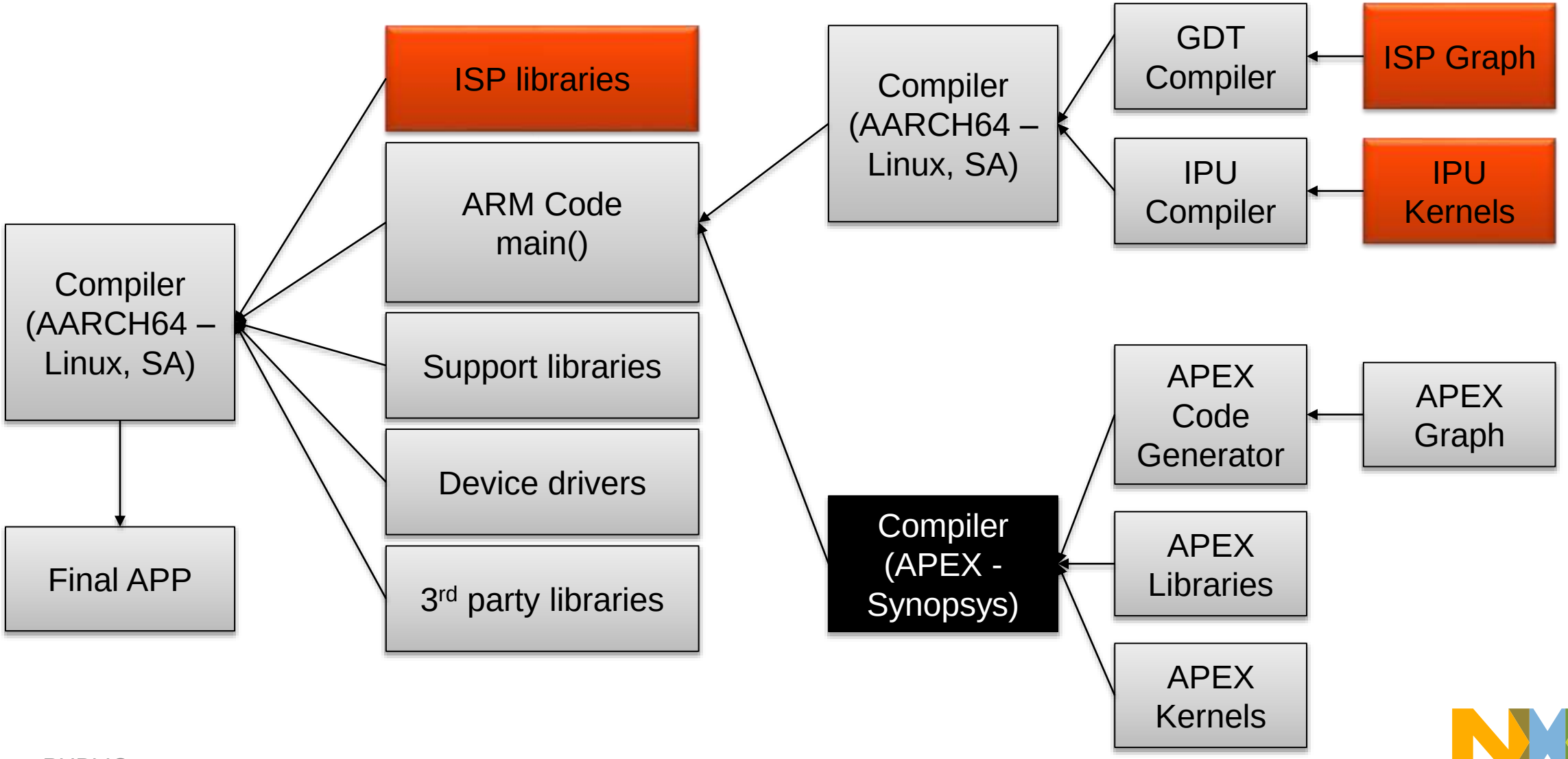
- **Common**
  - Contains some utility functions, like time measurement, C++ template for APEX process etc.
- **Log**
  - Logging library
- **Matrix**
  - Matrix structure definitions library
- **Neon**
  - NEON accelerated code (still under construction)
- **OAL**
  - OS Abstraction layer
- **OCL**
  - OpenCL library wrapper for APEX (C++ interface)

Provided in the VSDK with source code

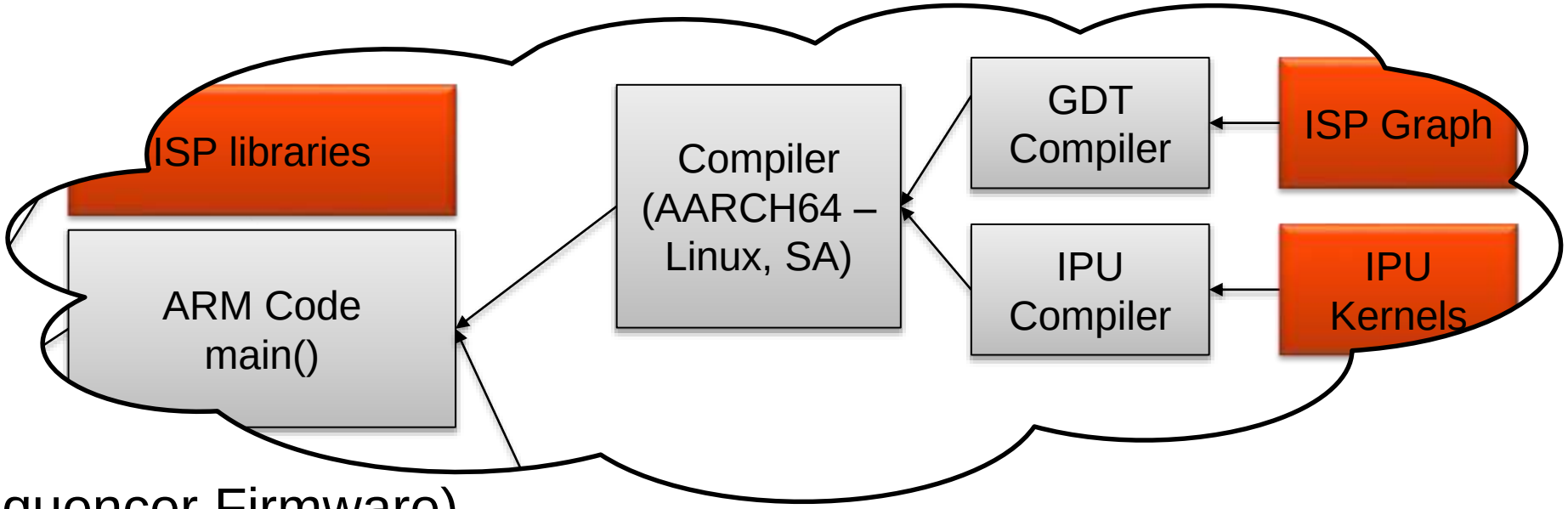
APEX related libs
APEXCV Base
APEXCV Pro
I/O
ISP related libs
Startup (for Standalone)
Utils (Auxiliary utility libs)



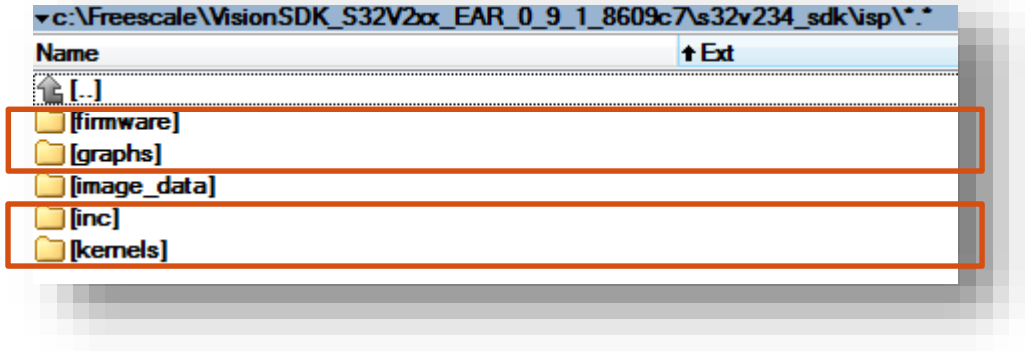
1.4 ISP Related resources



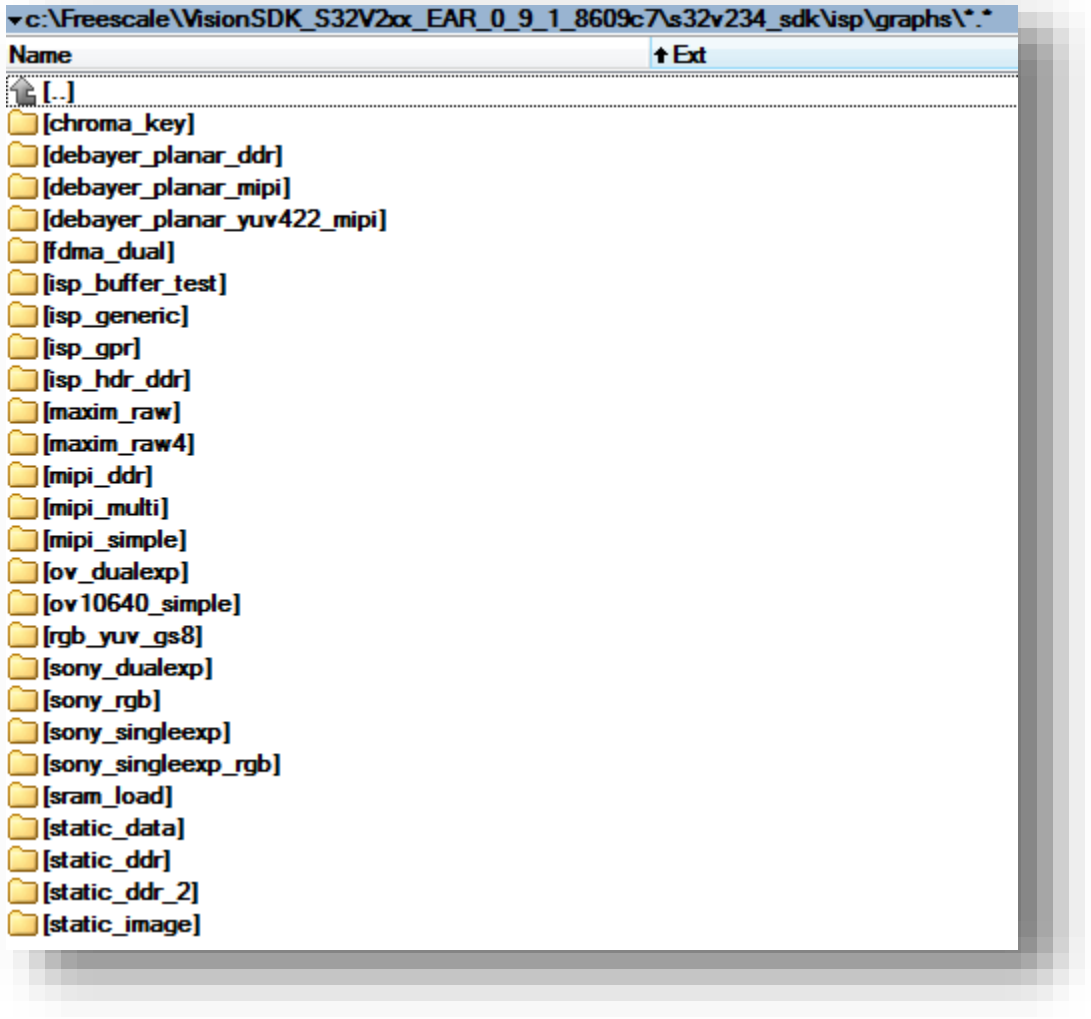
1.4 ISP Related resources

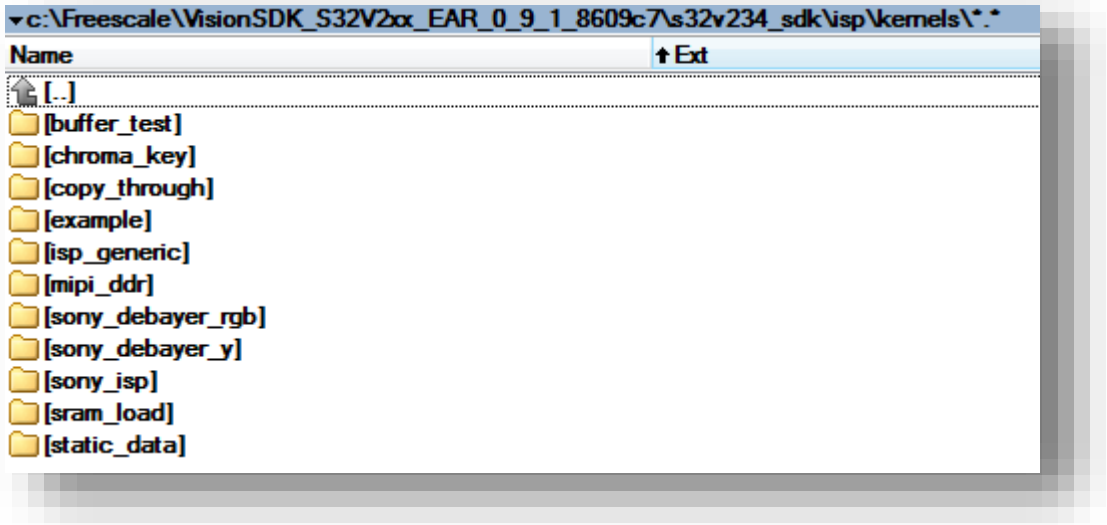


- ISP Libraries (Sequencer Firmware)
  - pre-compiled firmware
- ISP Graphs
  - Library of ISP Graphs to be used in application
- IPU Kernels
  - Library of IPU Kernels to be used in the graphs

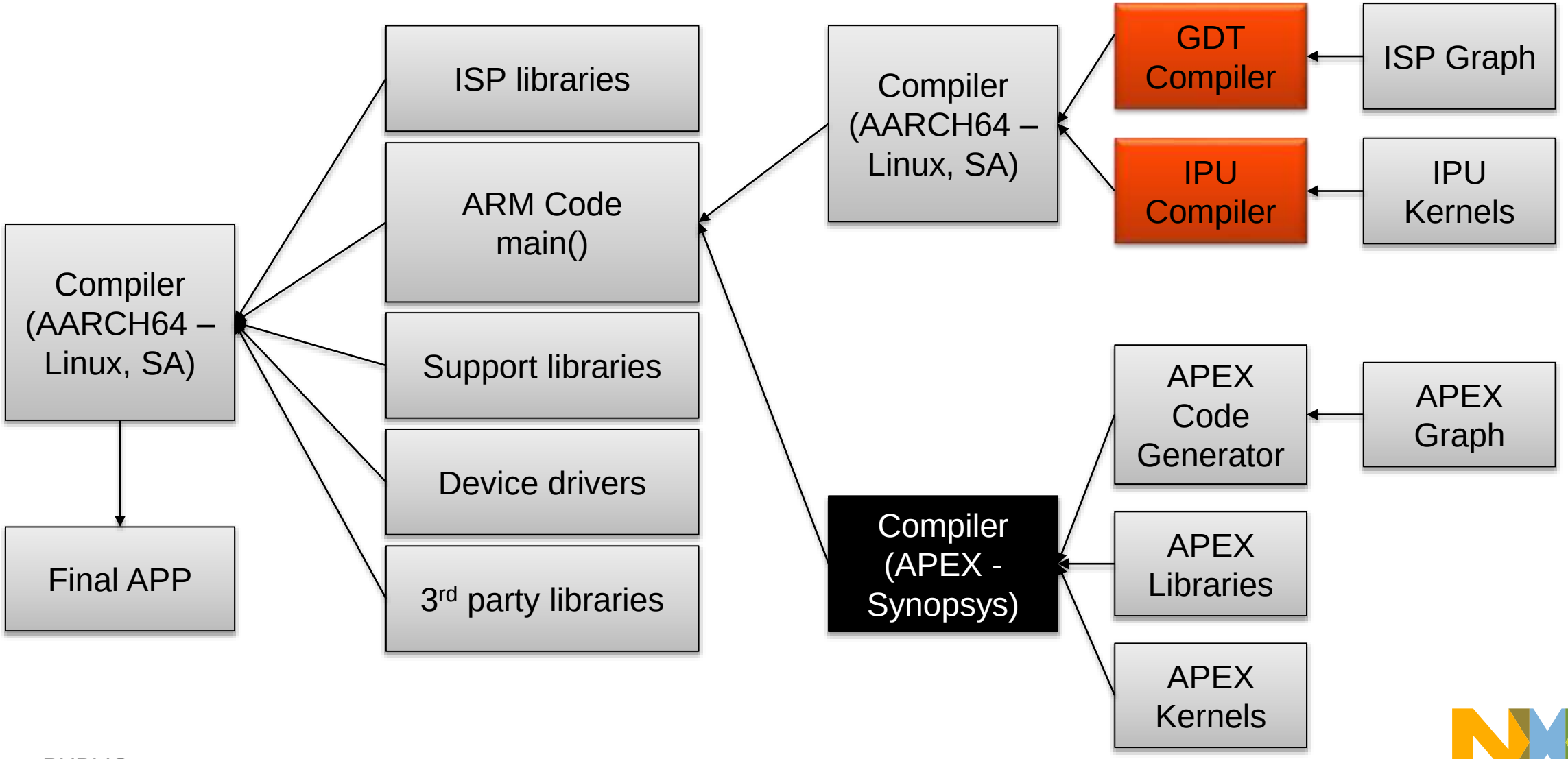


1.4 ISP Kernels & Graphs



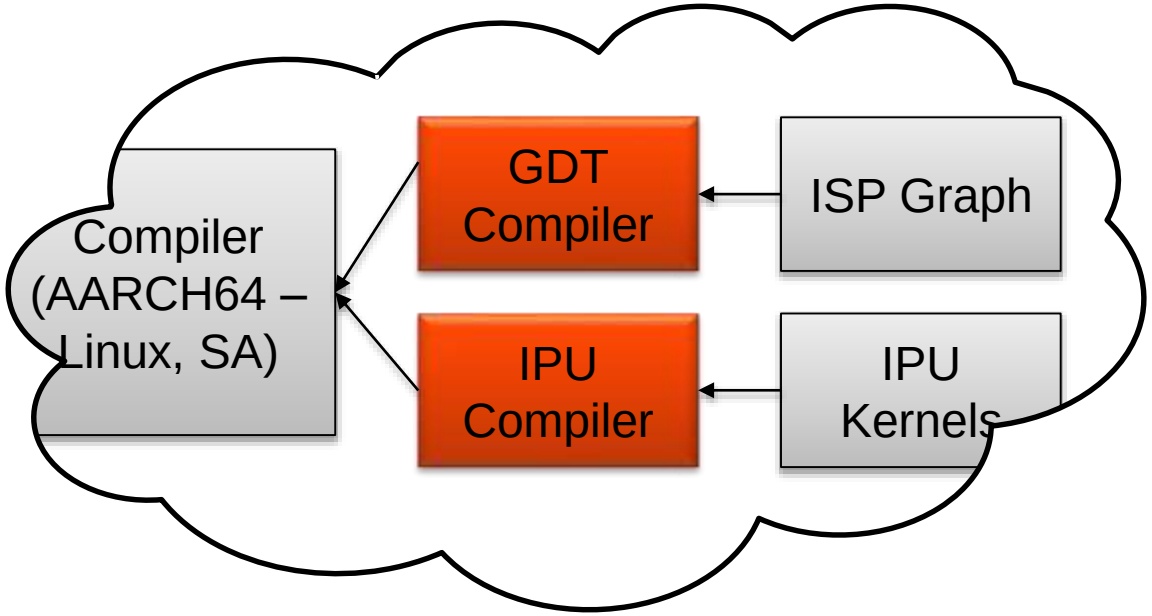


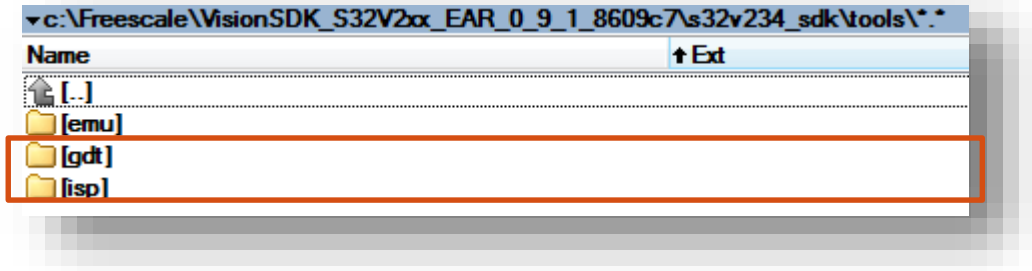
1.4 ISP Related resources



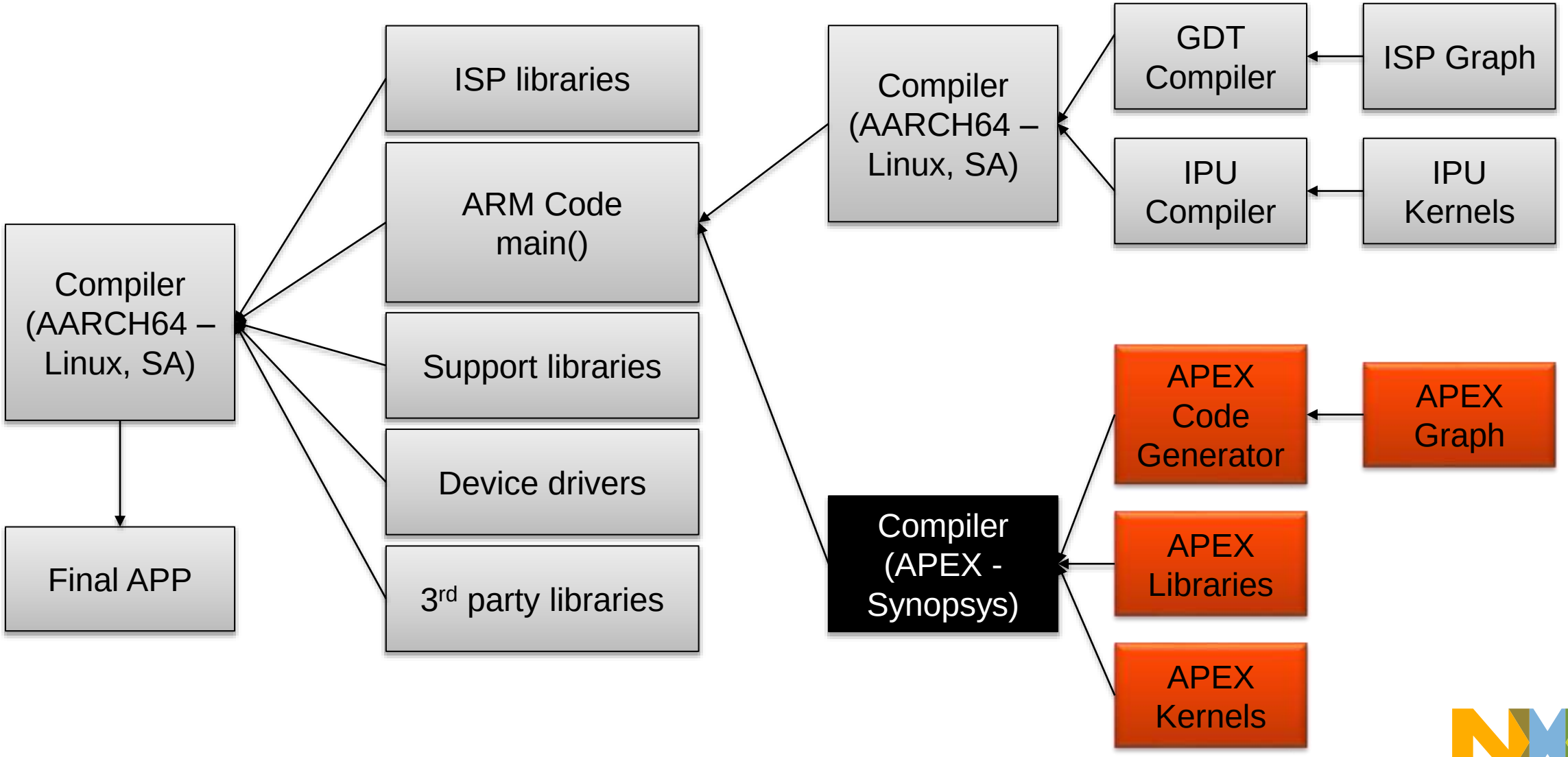
1.4 ISP Related Compilers

- Compilers used for ISP Graph & Kernels
  - GDT Compiler
    - Pre-compiled exe file used for ISP graph compilation
  - IPU Compiler
    - Pre-compiled exe file used for ISP kernel compilation





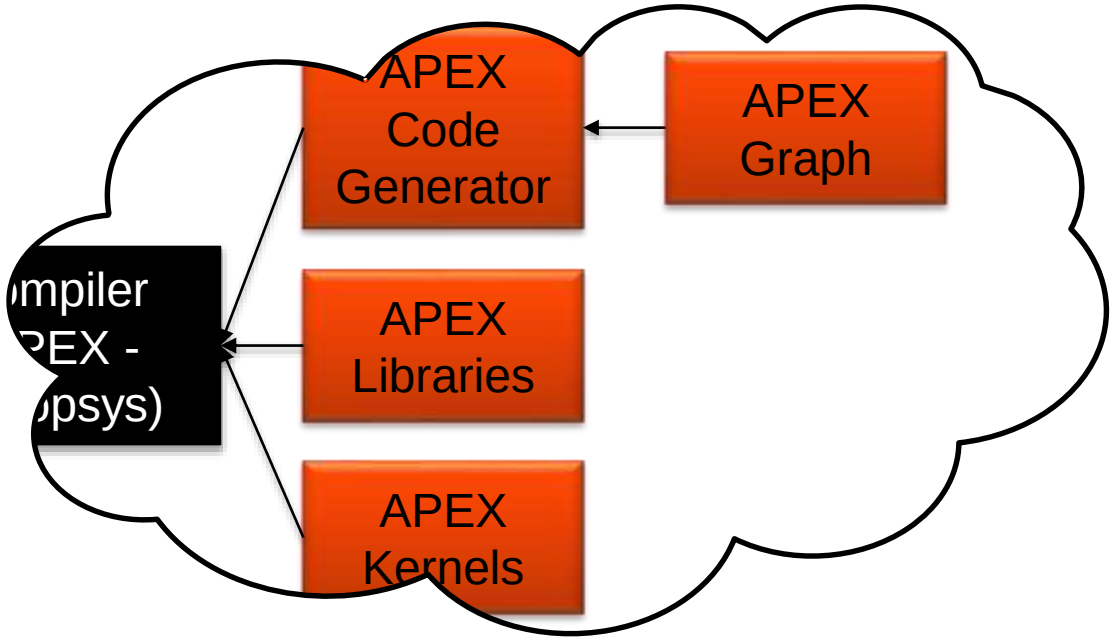
1.5 APEX Related resources

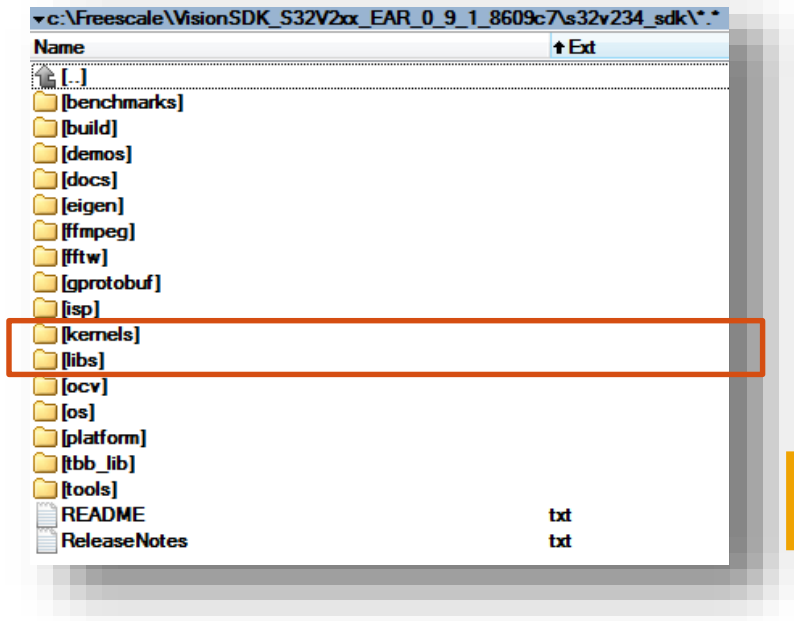


1.5 APEX Related resources

- APEX Code generator
  - Generates code for specific graph
  - **Source code not provided**
- APEX Graph
  - Graph definition for specific application
- APEX Libraries
  - Pre-compiled libraries with APEX firmware
  - **Source code not provided**
- APEX Kernels
  - APEX kernel library

- **APEX Compiler is not part of the VSDK**

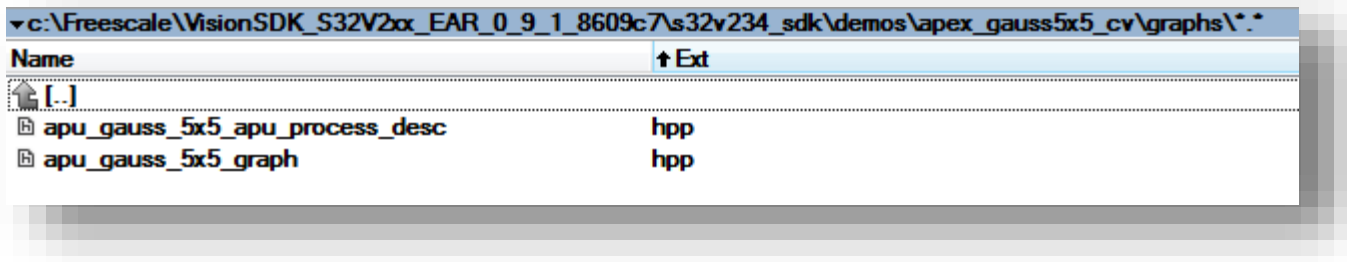




1.5 APEX Related resources – APEX Graph definition

- Application specific
- Header files - in each demo directory
  - Generated via graph tool or written by hand
- Automatically built with the application

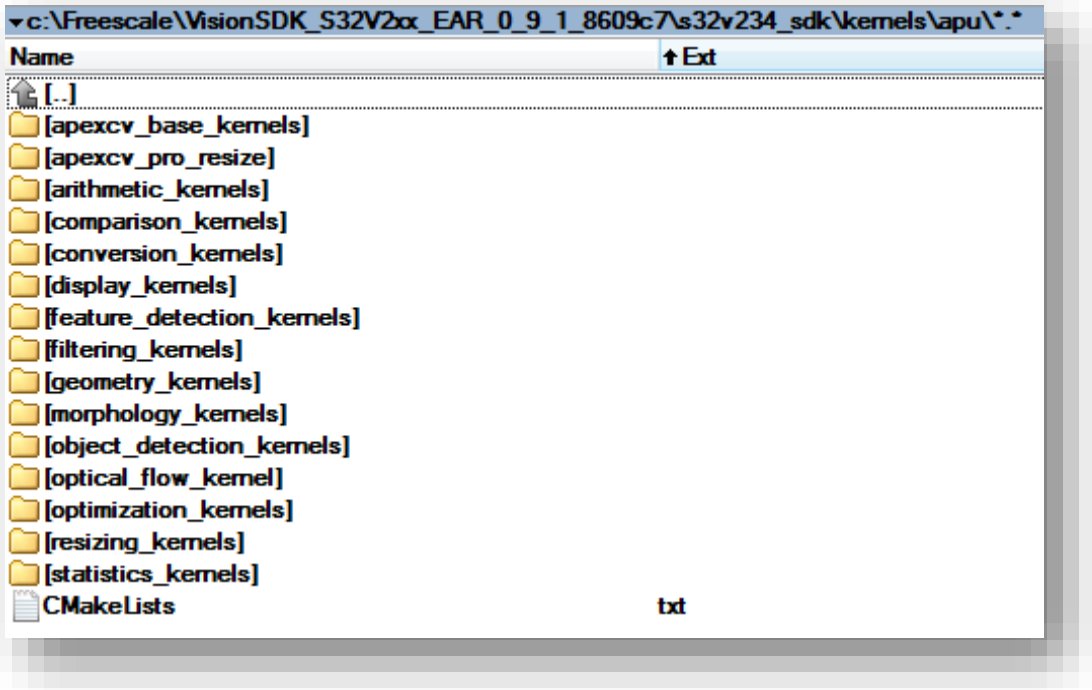
APEX Related resources
- **APEX Graphs**
- APEX Kernels
- APEX Firmware



1.5 APEX Related resources – APEX Kernel libraries

- Large amount of APEX kernels available
  - **APEXCV Base related**
  - **APEXCV Pro related** (if available in the release version)
  - **Arithmetic**
  - **Comparison kernels**
  - **Display kernels**
  - **Feature detection**
  - **Filtering**
  - **Geometry**
  - **Morphology**
  - **Object detection**
  - **Optical flow related**
  - **Optimization**
  - **Resize**
  - **Statistics**

APEX Related resources
- APEX Graphs
- **APEX Kernels**
- APEX Firmware

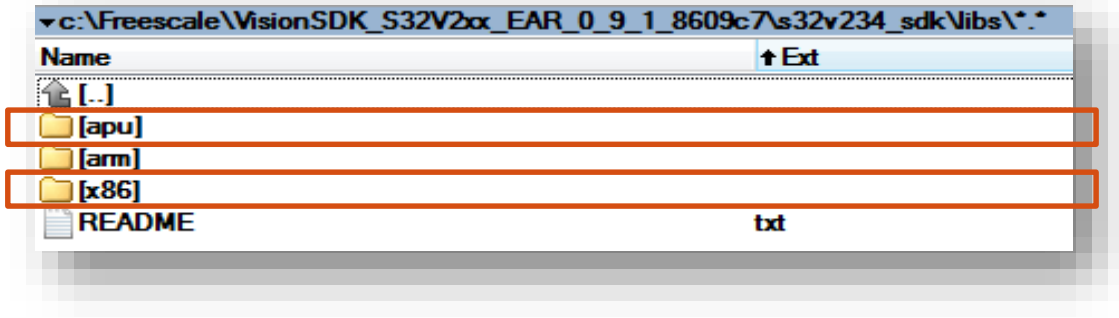


1.5 APEX Related resources – APEX Firmware

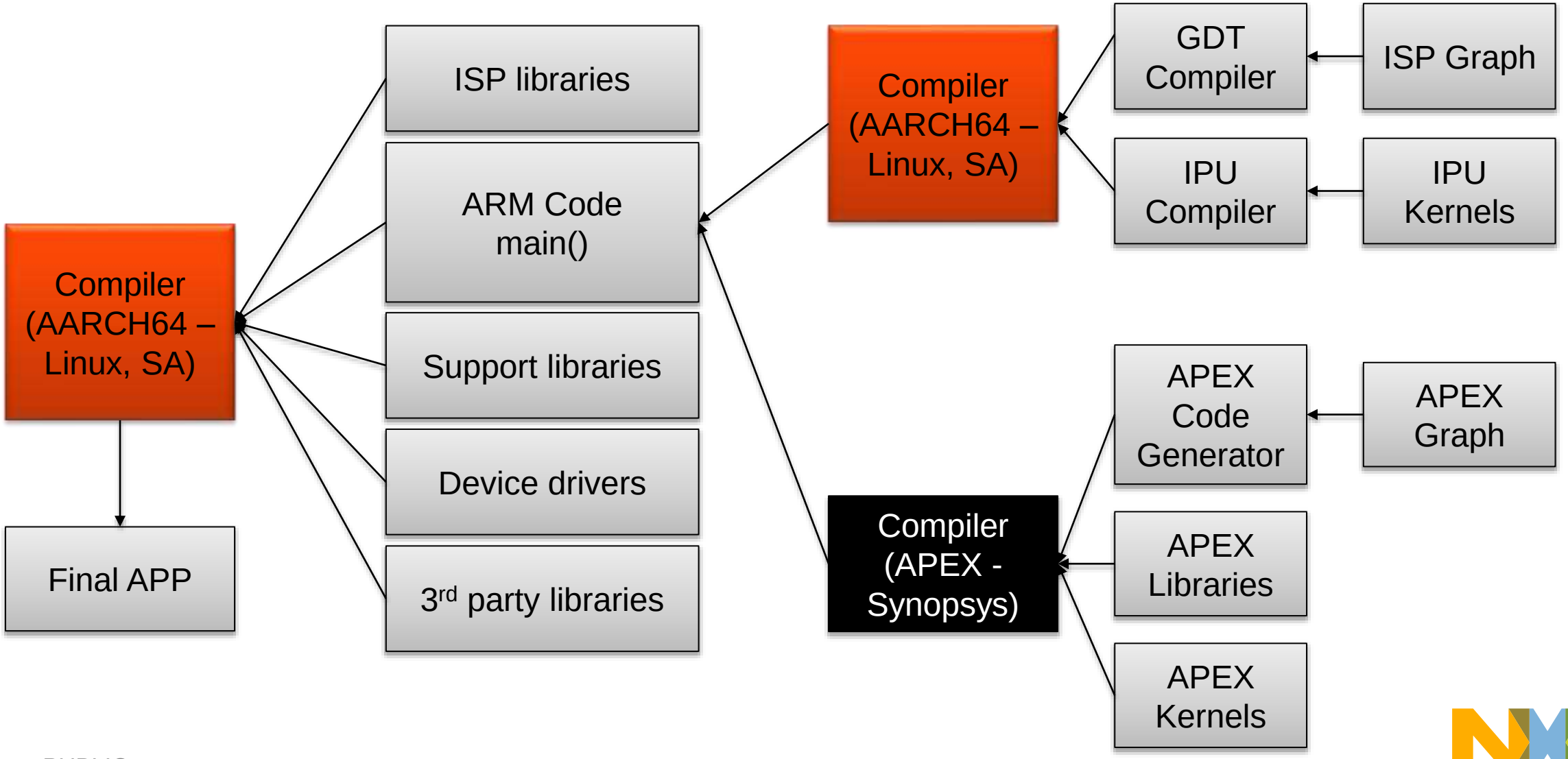
- Pre-compiled coded linked to the APEX Graph
- Contains APEX Firmware (APU)

- Contains APEX Code generator from graph Definitions (X86)
  - **The only platform specific code (Windows/Linux)**
  - If replaced by correct library, the VSDK content is platform interchangeable

APEX Related resources
- APEX Graphs
- APEX Kernels
- **APEX Firmware**



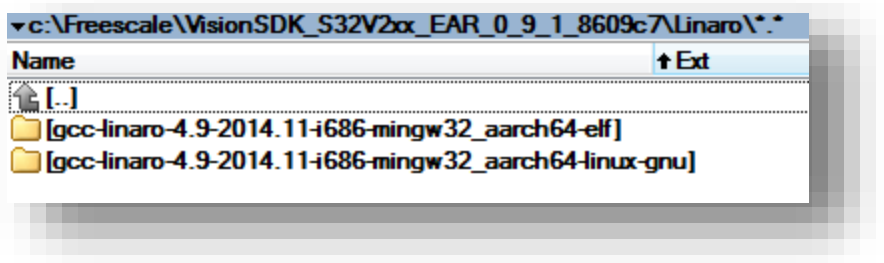
1.6 Compilers available



1.6 Compilers available

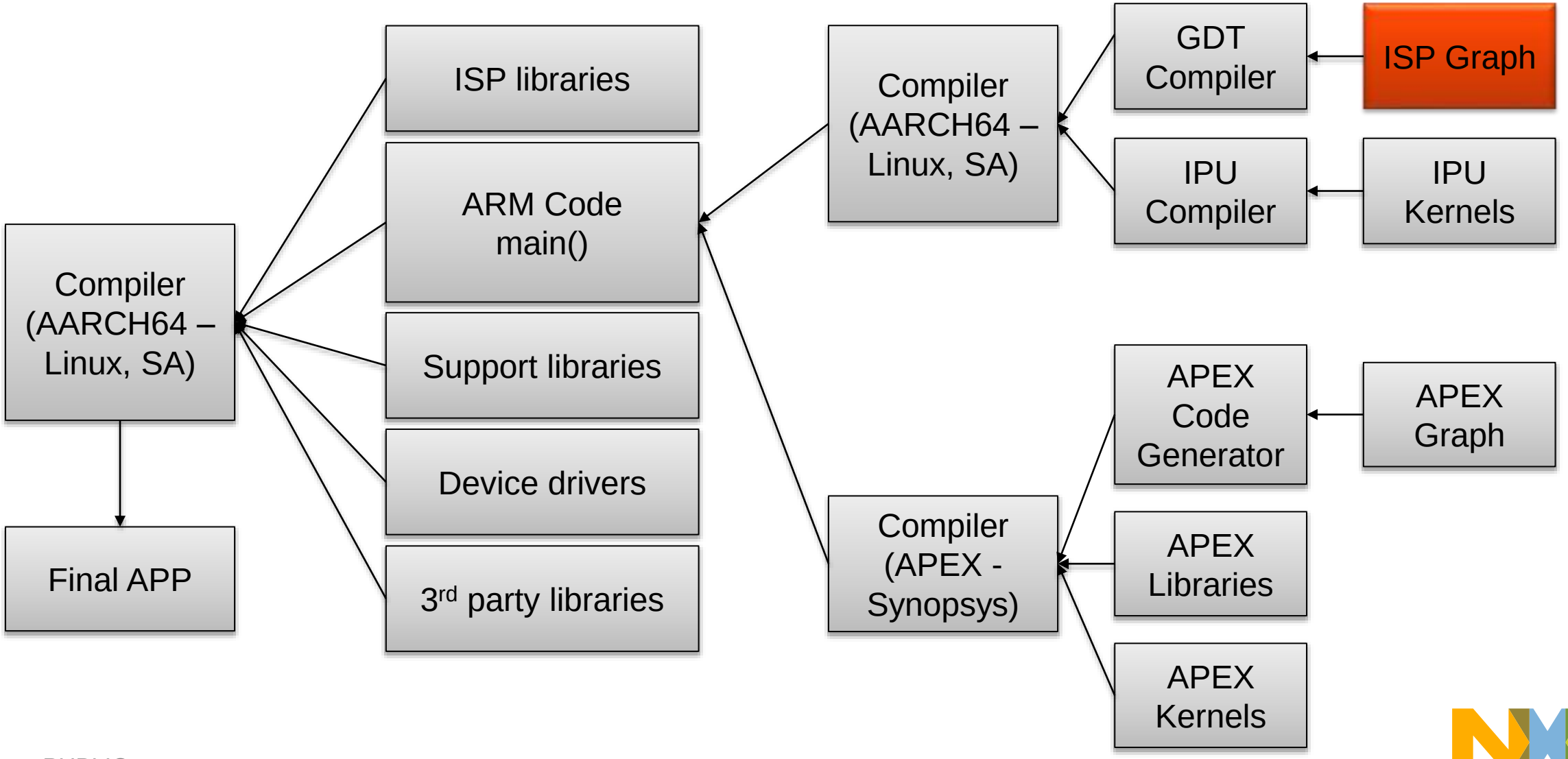
- Both AARCH64 Compilers are available:
  - Standalone application (bare metal)
  - Linux application

  - Installation is optional during the VSDK install
  - Paths set in the Windows environment by installer



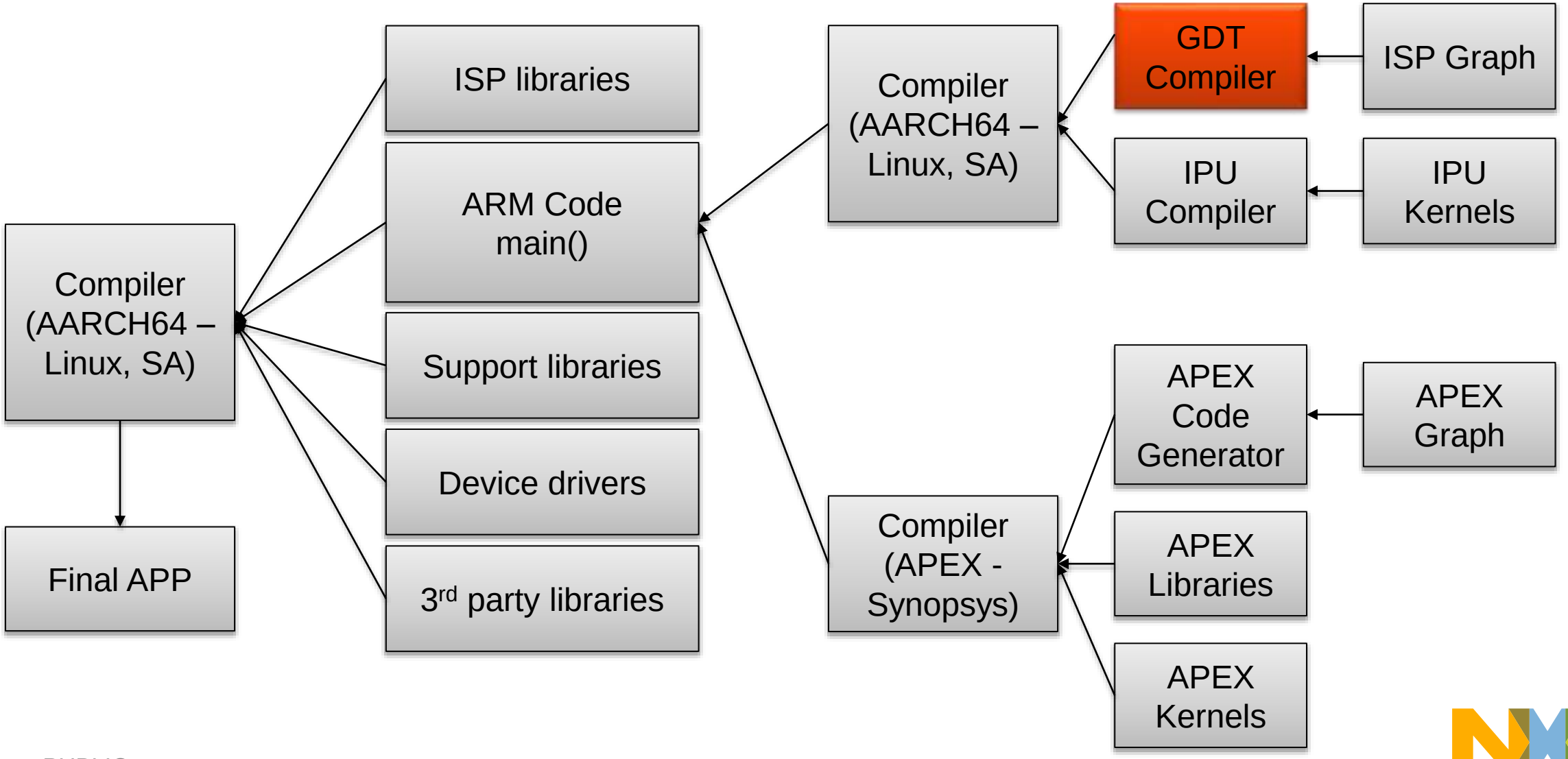
1.7 Build system – build sequence

Graph defined via graph design tool

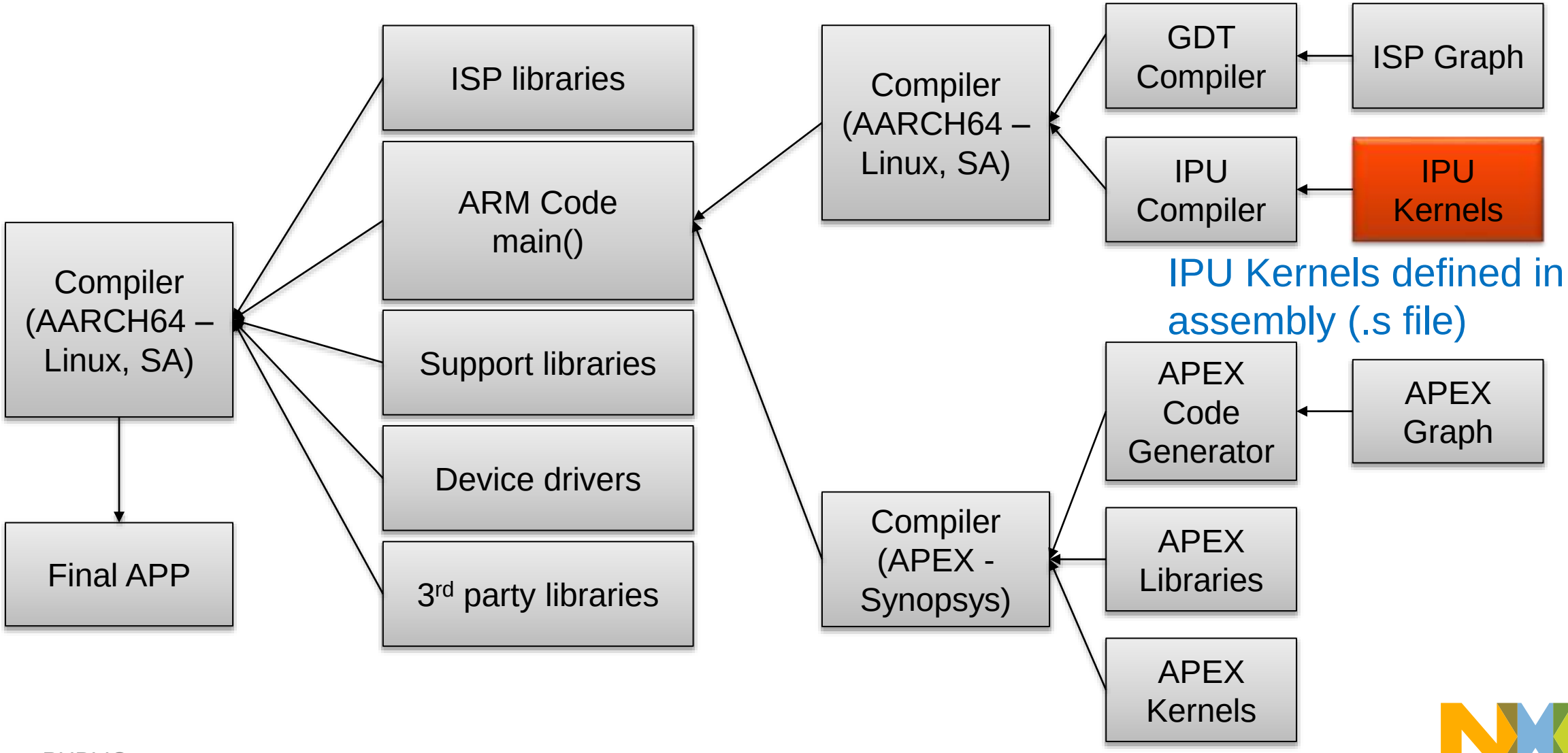


1.7 Build system – build sequence

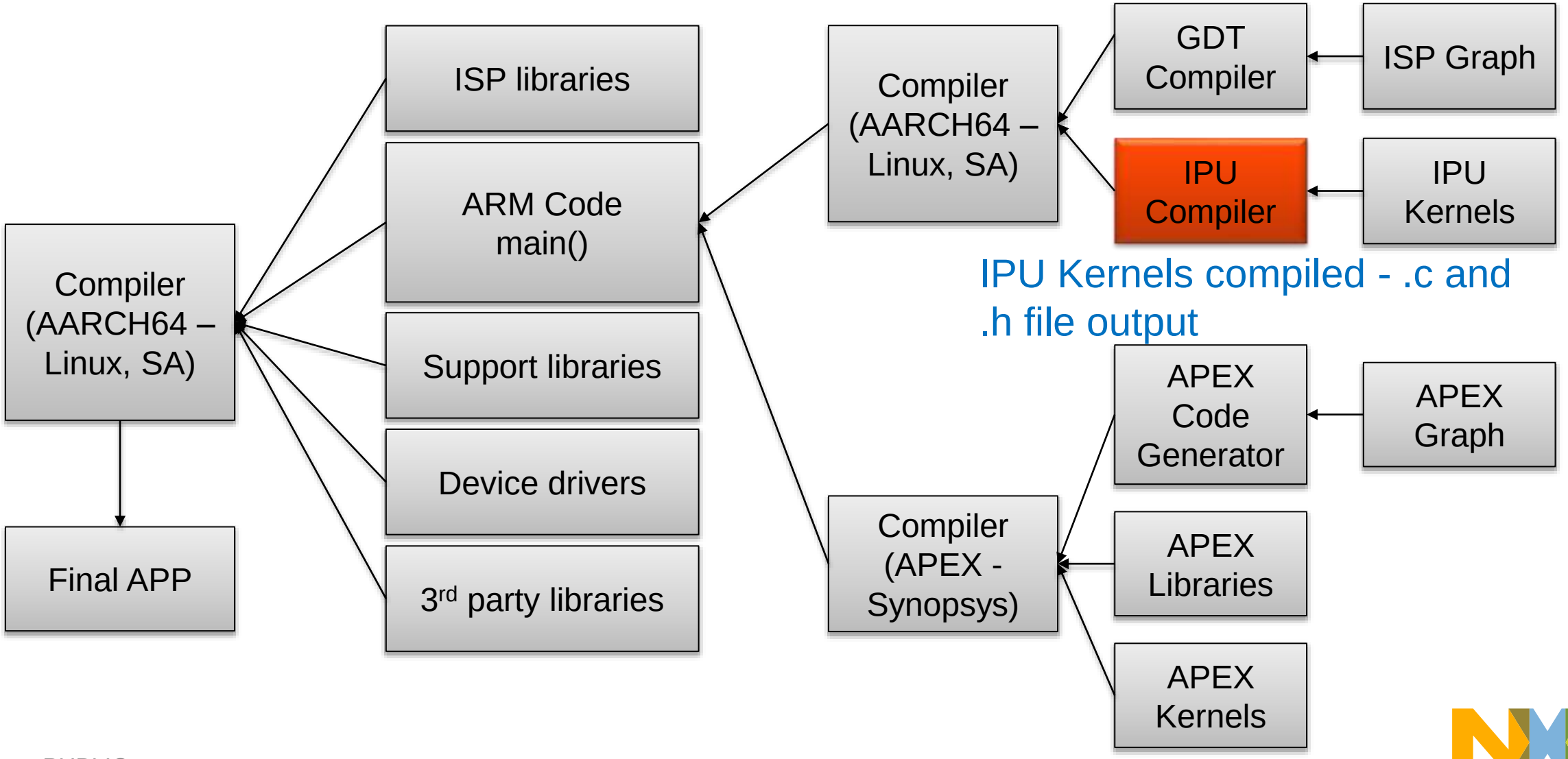
Graph compiled via GDT compiler -> .h and .c output



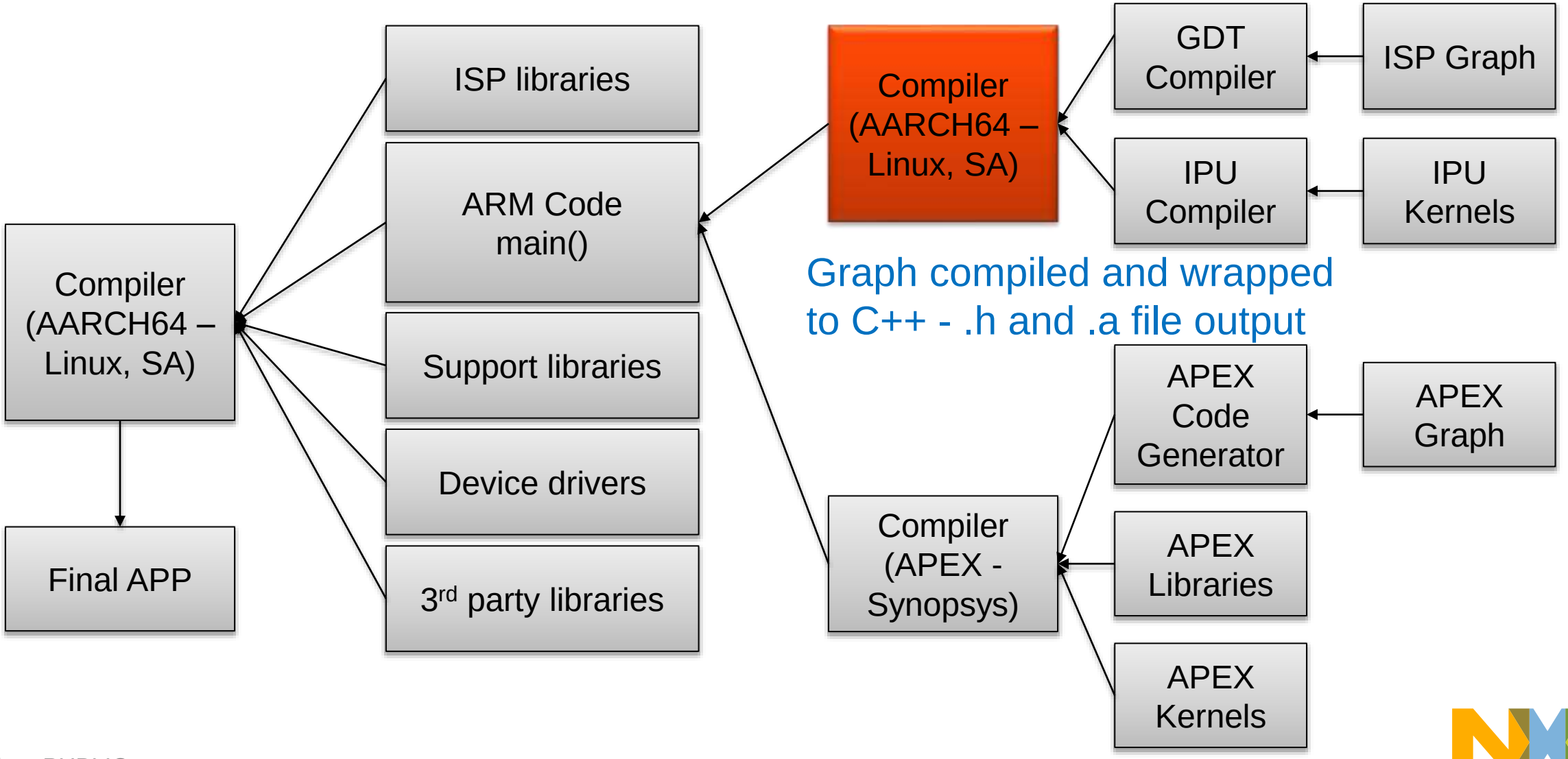
1.7 Build system – build sequence



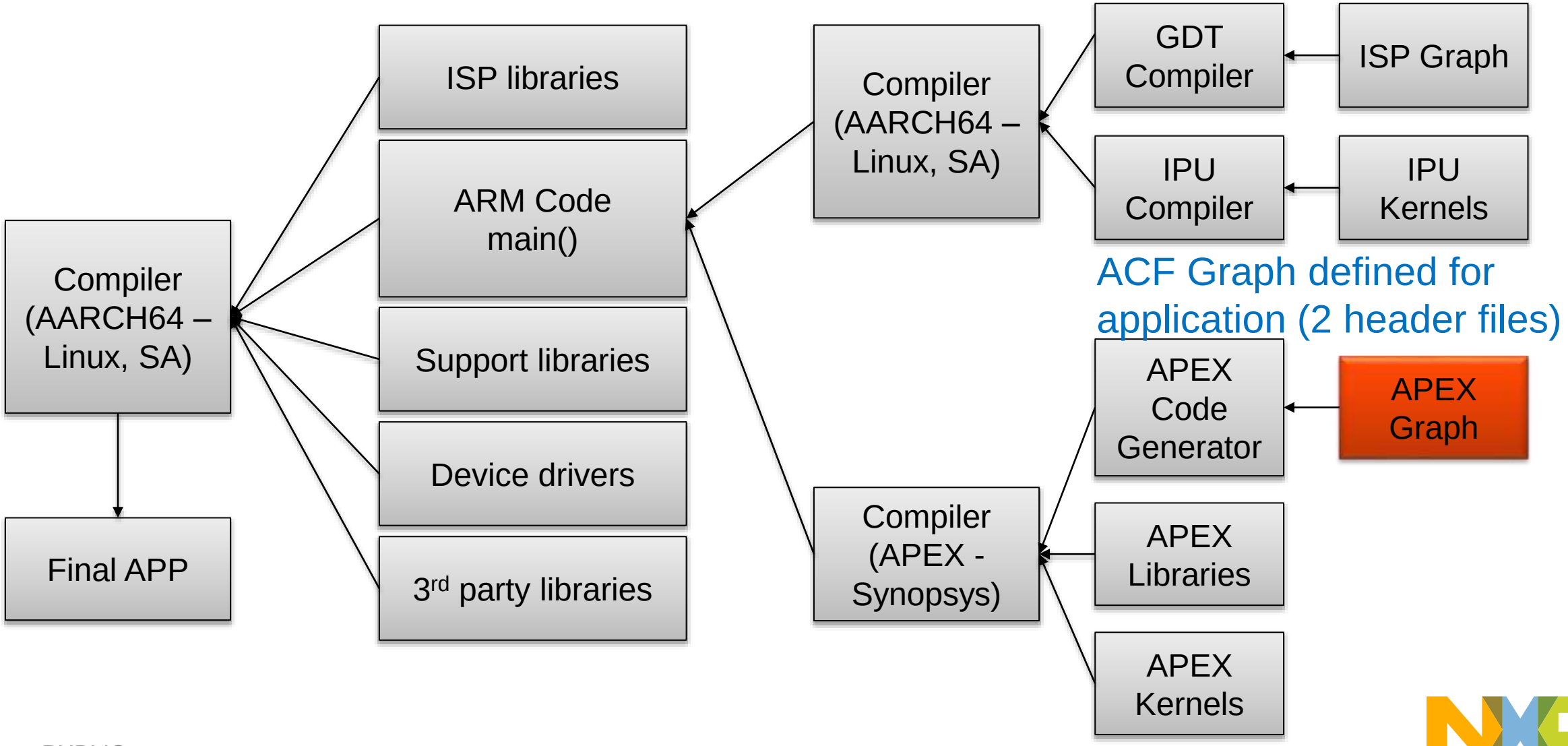
1.7 Build system – build sequence



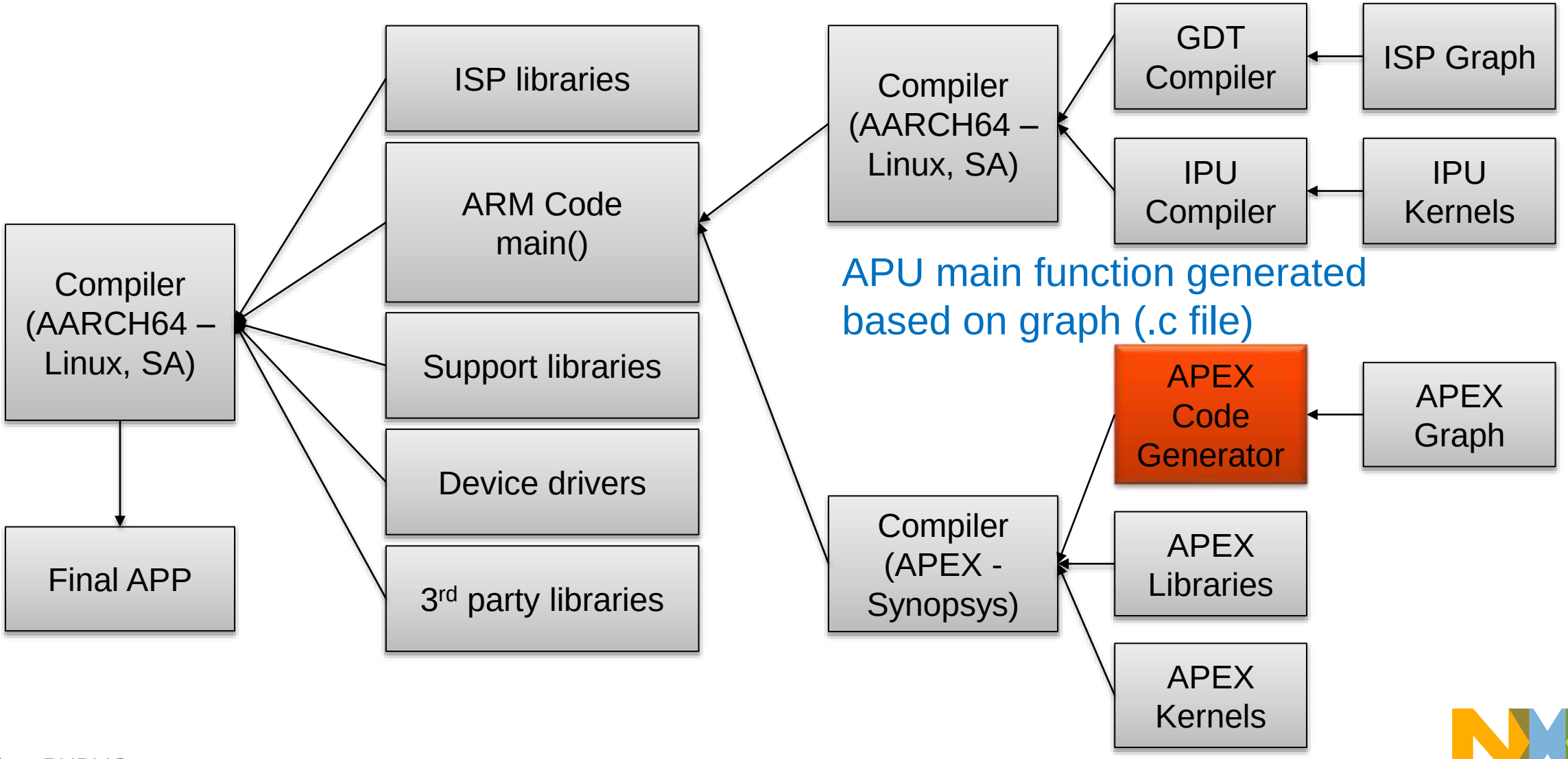
1.7 Build system – build sequence



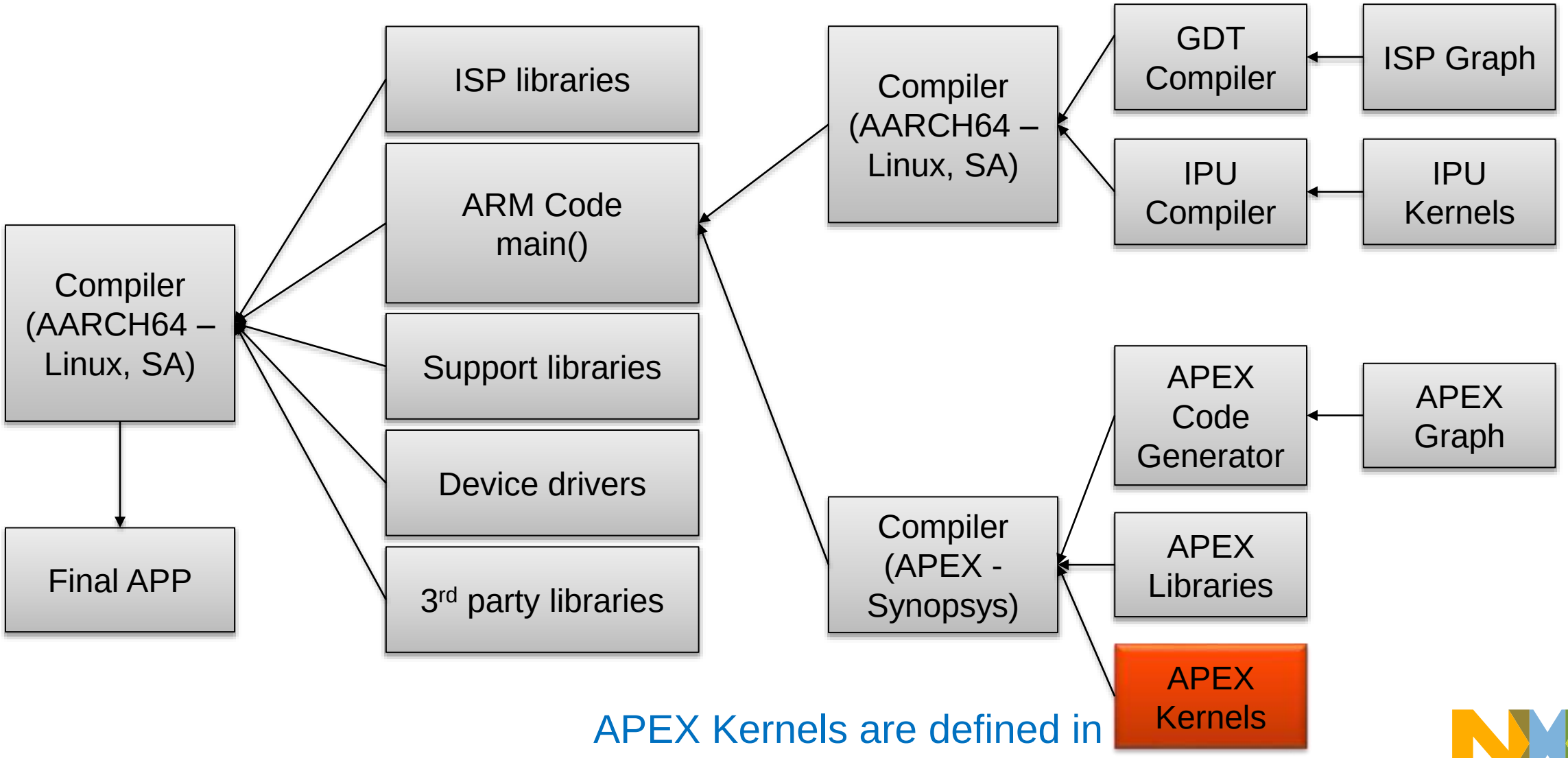
1.7 Build system – build sequence



1.7 Build system – build sequence

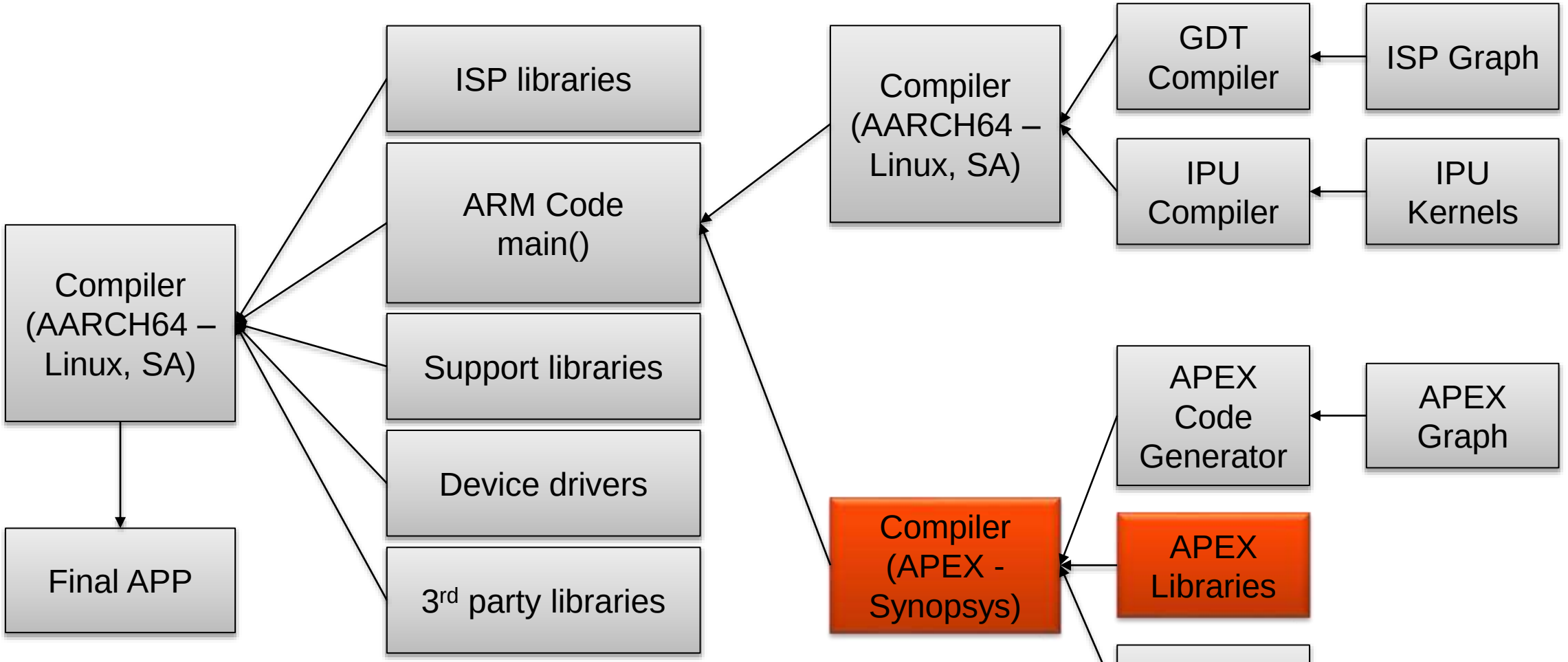


1.7 Build system – build sequence



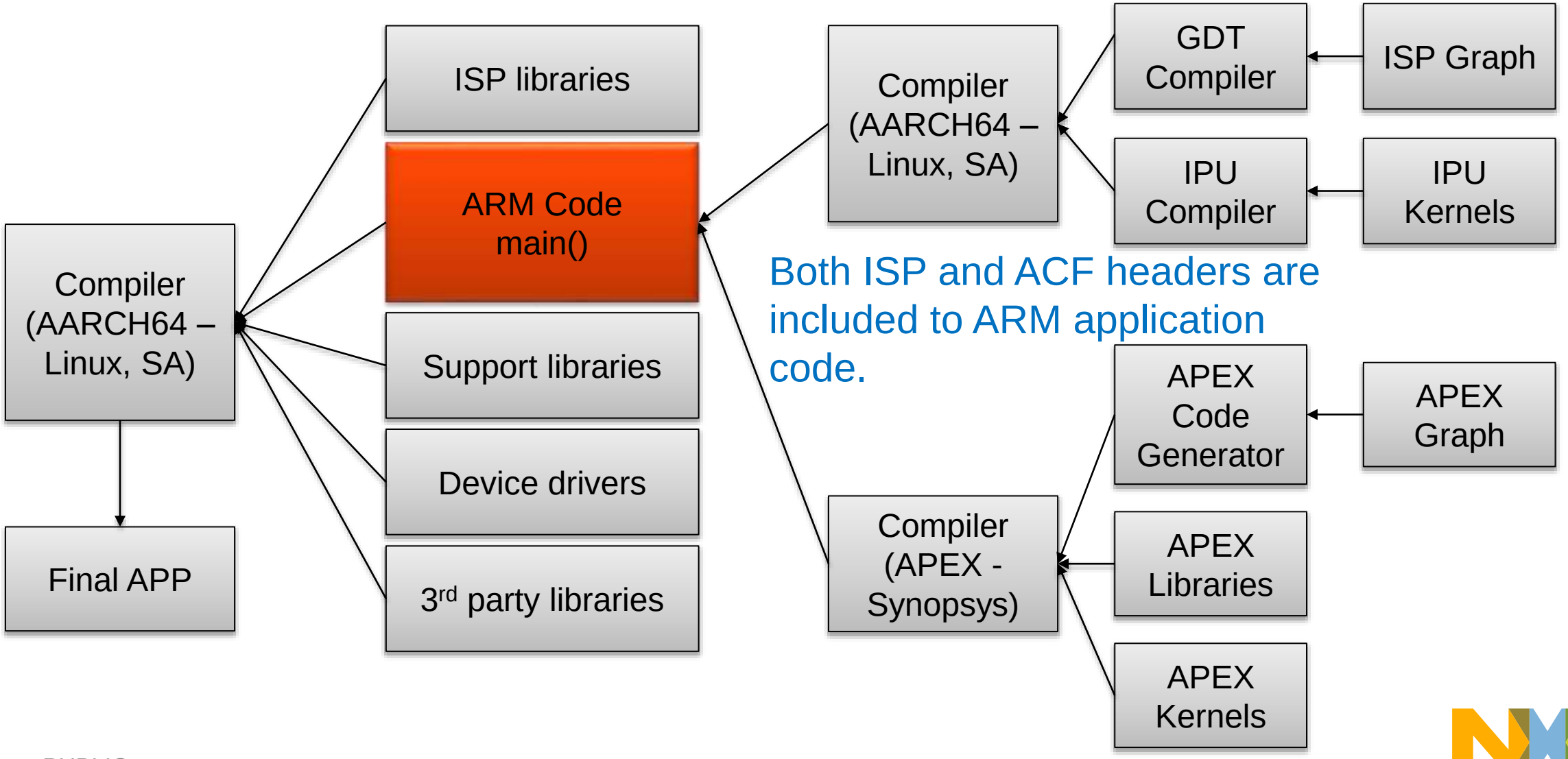
APEX Kernels are defined in extended C language

1.7 Build system – build sequence



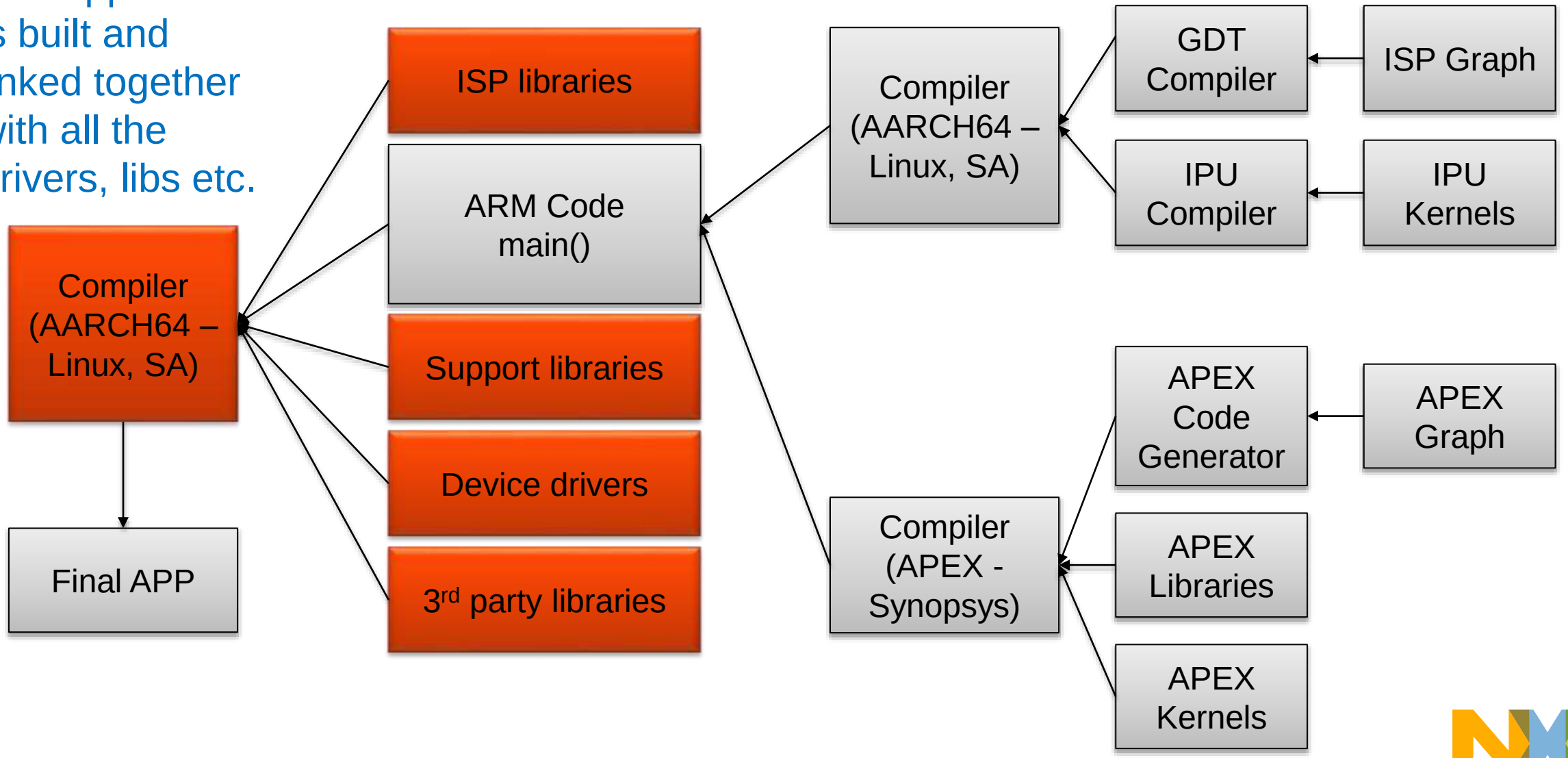
APU program compiled to .elf file + dumped to header file with C++ wrapper and program hexdump

1.7 Build system – build sequence

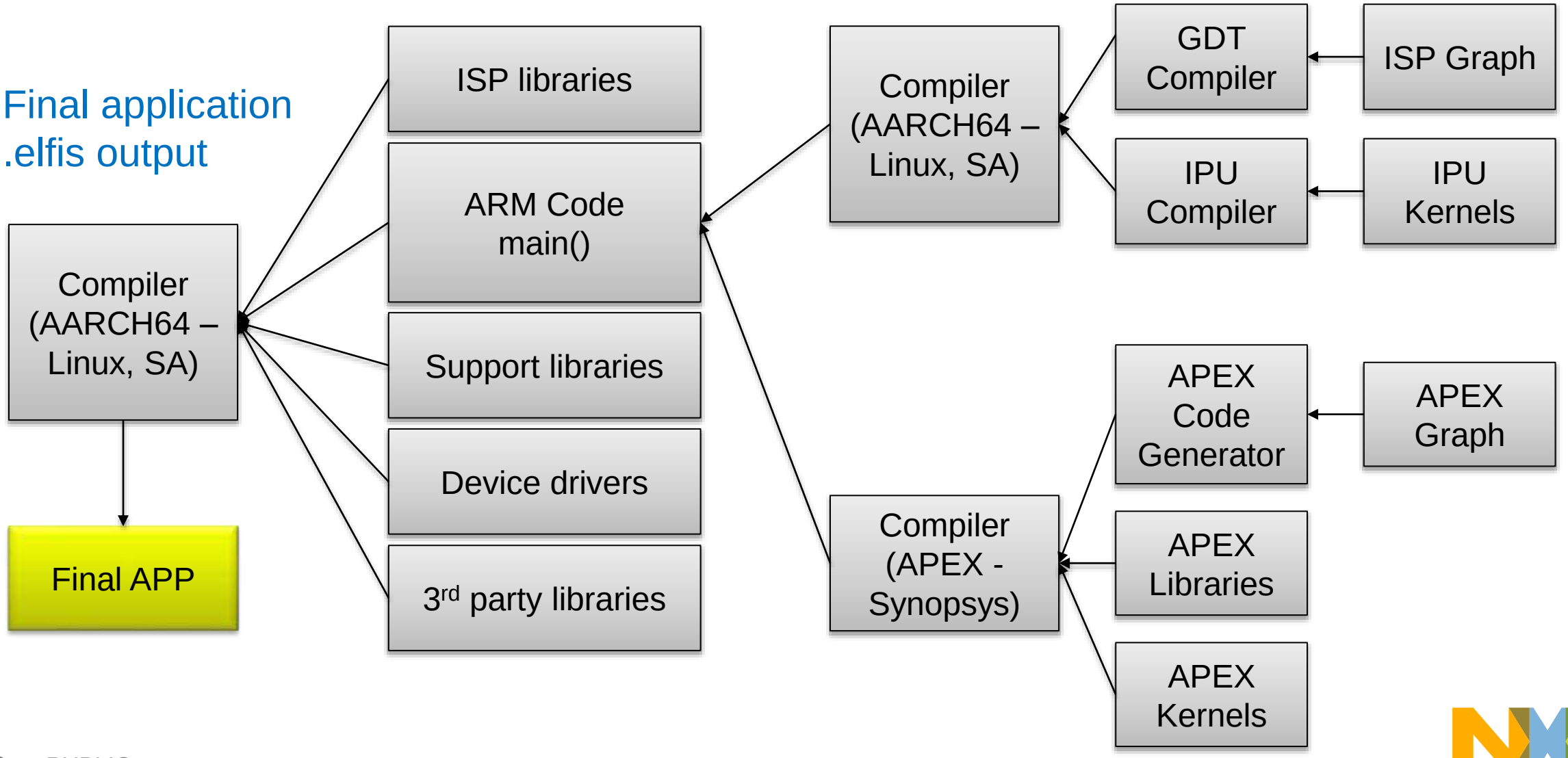


1.7 Build system – build sequence

Arm application is built and linked together with all the drivers, libs etc.



1.7 Build system – build sequence



1.7 Build system

- The whole build fully automatic in VSDK
- **Build invocation in target application directory builds everything**

- Each target directory contains various build directories according to platform:
  - build-v234ce-gnu-sa-d
  - build-v234ce-gnu-linux-d
  - build-v234ce-ghs-integrity-d
  - build-deskwin32

  - build-apu-tct-sa-d
  - build-x86-gnu-linux-d

1.7 Build system – building an app/lib

- Application/Library contains following build definition:
  - build-[target]-[compiler]-[platform]-[debug/optimized] directory with Makefile
  - BUILD.mk file with build definition

- BUILD.mk:
  - Specific to application/library
  - Specifies:
    - Sources
    - Linked libraries (if any)
    - Header file directories
    - ISP Definitions (graph used)
    - APEX Definitions (graph/kernel libs used)

1.7 Build system – building an app/lib

- Enter the build-* directory according to the target
  - **make allsub**
    - builds everything (APEX, ISP), checks for changes in dependencies
  - **make cleansub**
    - removes everything incl. APEX and ISP files, **keeps the .d cache files**
  - **make purgesub**
    - Removes .d cache files
  - **make APEX_PREBUILD=1 allsub**
    - Builds everything excl. APEX (if compiler is not present)
  - **make APEX_PREBUILD=1 cleansub**
    - Cleans everything excl. APEX files

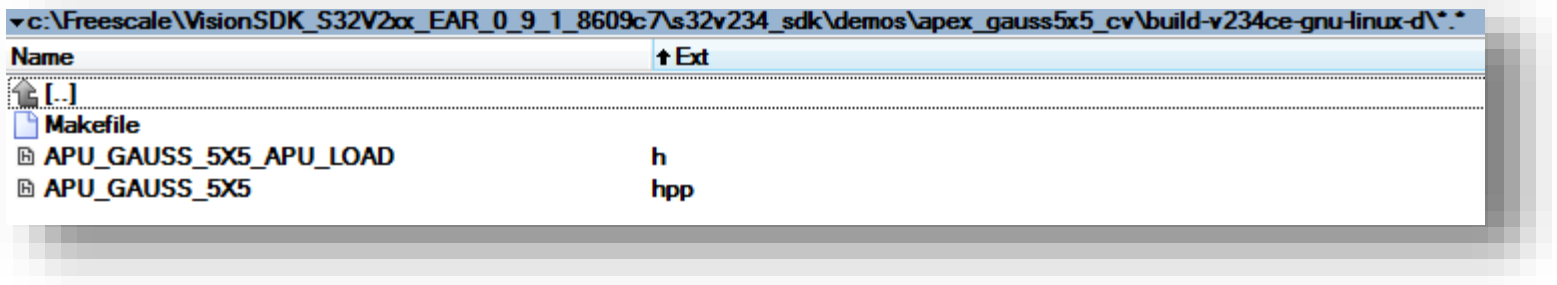
1.7 Build system – building an app/lib – pre-build APEX program

- VSDK is installed with pre-build APEX graphs. The user can use the APEX_PREBUILD define.

- **make clean** or **make cleansub** will delete those files permanently

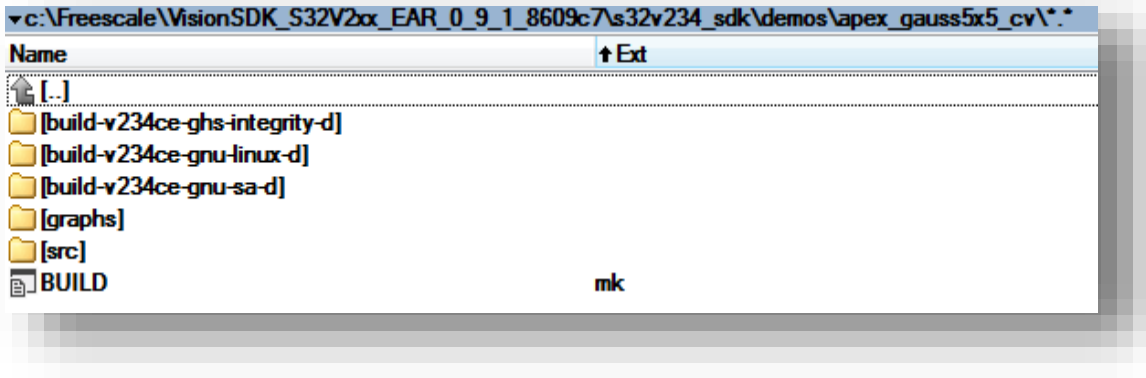
  -> **APEX_PREBUILD won't be useable anymore**

- **make APEX_PREBUILD=1 clean** or **make APEX_PREBUILD=1 cleansub** is imperative to be used



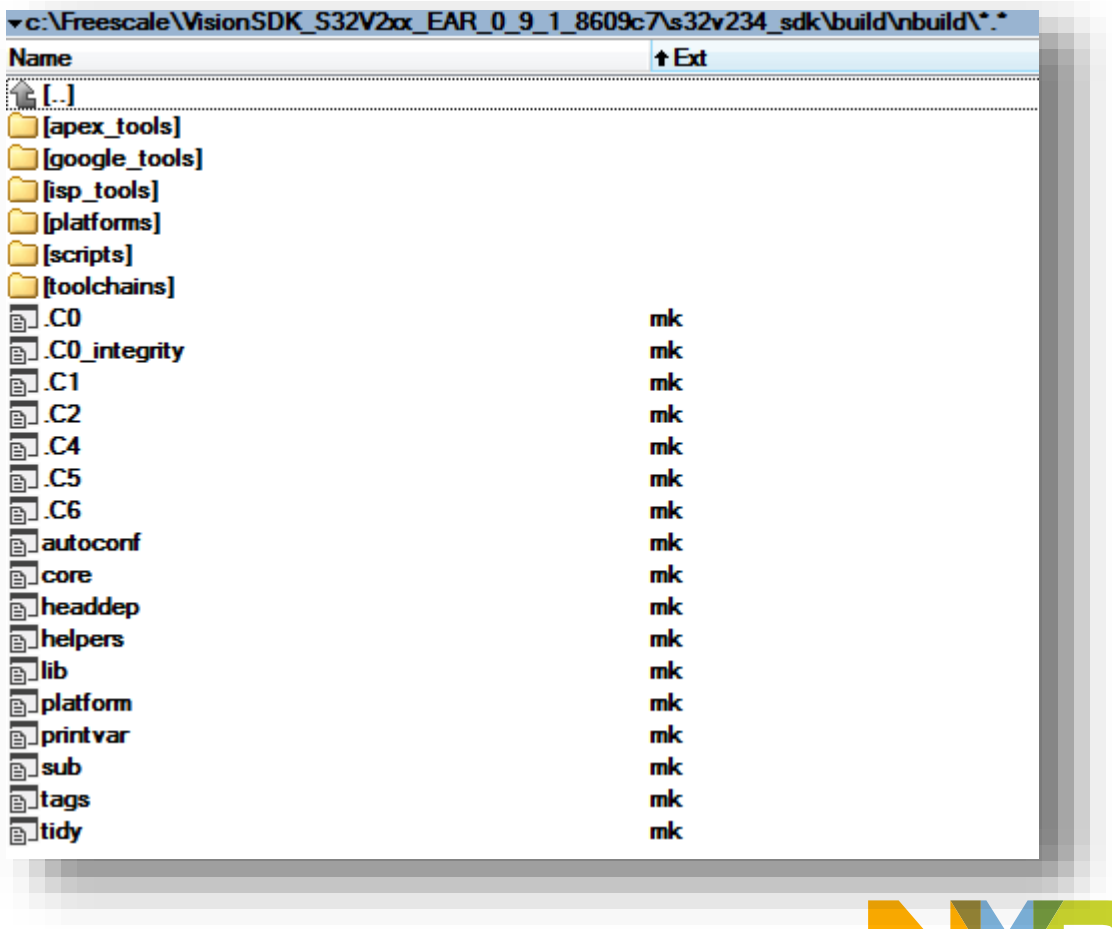
1.7 Build system – building an app/lib

- **Example application – s32v234_sdk/demos/apex_gauss5x5_cv/**
- Enter the build-* directory according to the target and execute the sequence:
  - **make APEX_PREBUILD=1 allsub**
    - builds everything, elf is output
  - **make clean**
    - removes everything **incl. APEX prebuilt headers**
  - **make purgesub**
    - resets the cache
    - when switching non-APEX and APEX build, it's obligatory
  - **make allsub**
    - builds everything incl. the APEX graph



1.7 Build system – advanced build defaults

- Default compilers, parameters etc. defined in s32v234_sdk/build/nbuild directory
- **platforms**
  - Contains specific Makefiles for platform
- **toolchains**
  - Contains compiler toolchains + parameters
- **apex_tools**
  - APEX related toolchains
- **isp_tools**
  - ISP related toolchains

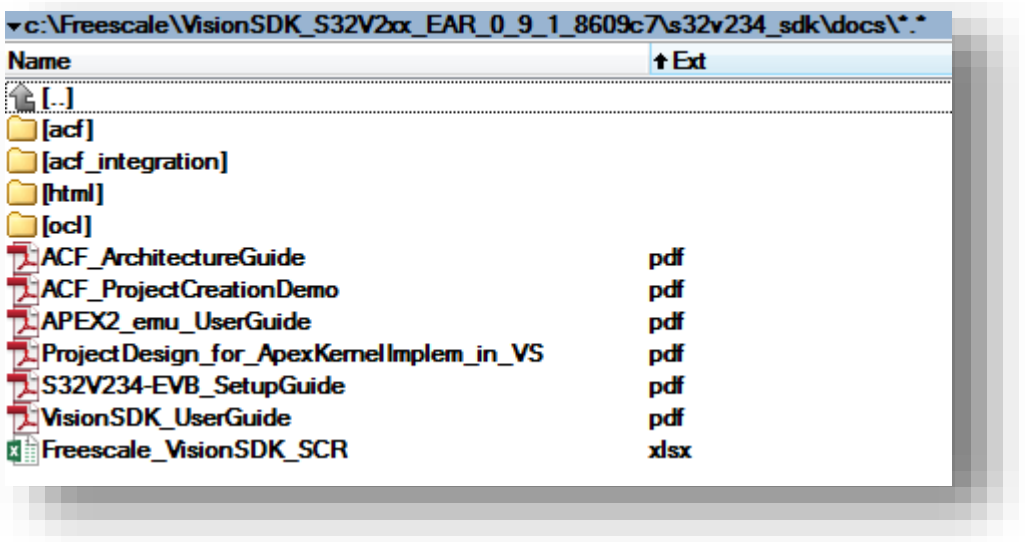


Part 2

1. Documentation
2. Demos
3. OS related content
4. Running the demos
5. Debugging

2.1 Documentation

- All available documentation is part of VSDK installation
- **acf** – ACF Programming and architecture overview
- **html** – Doxygen generated docu for APEX kernels
- **Ocl** – APEX OpenCL related documentation

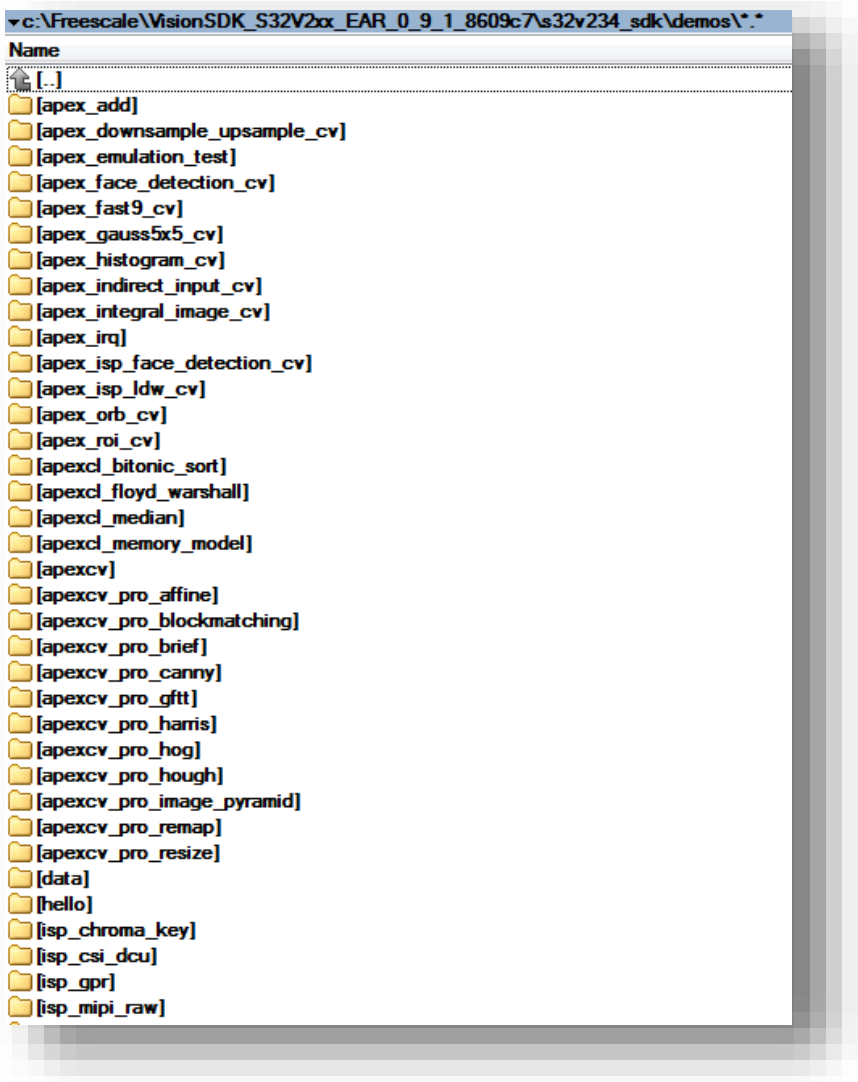


- **ACF Architecture Guide**
  - ACF Architecture integration in VSDK
- **ACF Project Creation Demo**
- **APEX2 EMU User Guide**
- **S32V234-EVB Setup Guide**
  - Setup of EVB (outdated now for REV D boards)
- **Vision SDK User Guide**
  - VSDK Main User Guide – contains description of prerequisites, build, demos etc…

2.2 Demos

- Demo applications are available in VSDK
  - Aimed to describe simple functionality

- Available in s32v234_sdk/demos directory
  - Name contains main demo target
    - apex
    - isp
    - neon
    - cv



2.2 Demos – list – part 1

- **apex_add**
  - simple addition demo, two APEX devices in parallel, C++ template wrapper used for ACF process
- **apex_downsample_upsample_cv**
  - resize demo on APEX
- **apex_emulation_cv**
  - emulation library example – buildable on windows or for target
- **apex_face_detection_cv**
  - Face detection demo on APEX, ARM (NXP and OpenCV)
- **apex_fast9_cv**
  - Fast 9 corner detection demo on APEX
- **apex_gauss5x5_cv**
  - Gauss filtering on APEX
  - **Simplest demo for APEX**

2.2 Demos – list – part 2

- **apex_histogram_cv**
  - Histogram computation on APEX
- **apex_indirect_input_cv**
  - Rotate image on APEX (using ACF indirect input)
- **apex_integral_image_cv**
  - Integral image computation and filtering on APEX
- **apex_irq**
  - APEX application asynchronous run
- **apex_isp_face_detection_cv**
  - APEX Face detection with camera input through ISP
- **apex_isp_ldw_cv**
  - APEX Lane Departure Warning System with camera input through ISP

2.2 Demos – list – part 3

- **apex_orb_cv**
  - ORB Image matching on APEX
- **apex_roi_cv**
  - APEX ROI functionality and use
- **apexcl_***
  - OpenCL for APEX demos
- **apexcv**
  - APEX CV base example (all algorithms available)
- **apexcv_pro_***
  - APEX CV base example (all algorithms available)
- **hello**
  - Hello world application

2.2 Demos – list – part 4

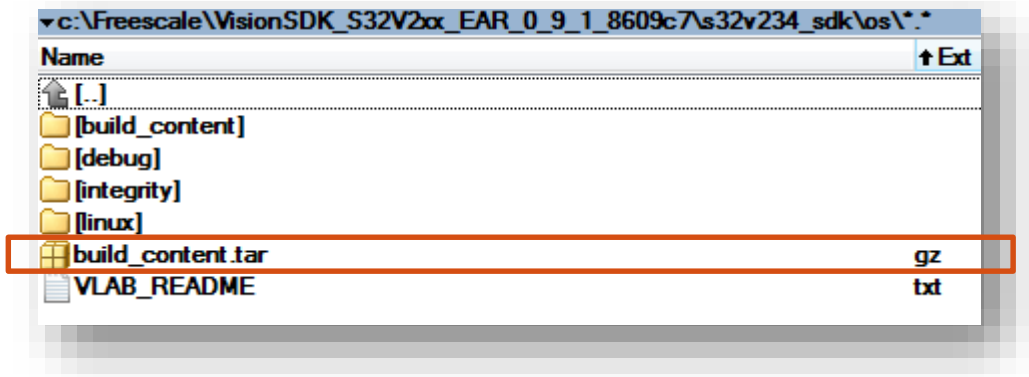
- **isp_chroma_key**
  - Mapping of specific color range to blue or green via ISP
- **isp_csi_dcu**
  - Simple camera input through ISP and output on display
  - **Simplest demo for ISP**
- **isp_gpr**
  - Parametrization of IPU processing at runtime
- **isp_mipi_raw**
  - Raw data from camera output through ISP
- **neon_eigen**
  - Simple matrix addition using Eigen library accelerated by ARM Neon
- **neon_fftw_cv**
  - Forward/Backward Fourier transform using FFTW library accelerated by ARM Neon

2.2 Demos – list – part 5

- **isp_gauss3x3_cv**
  - Gaussian filter implemented on ARM Neon
- **qspi_readwrite**
  - SD Card boot and QSPI read-write interface for standalone

2.3 OS Related Content

- Pre-built OS environment is available in VSDK – s32v234_sdk/os/build_content.tar.gz:
  - **v234_baremetal_build**
    - All demos pre-built
    - Lauterbach script is available for load of each demo
    - .bin file is built to be loaded on SD card
  - **v234_linux_build**
    - BSP 5.1 prebuilt and available for SD Card load (uImage, s32v234-evb.dtb, u-boot.s32, rootfs.tar)
    - All demos pre-built in the root file system on SD Card
    - README.txt is available with SD Card creation information



2.4 Running the demos - Standalone

- Every demo contains following files after build (build-v234ce-gnu-sa-d):

  ***.elf file**

  - Executable to be loaded via Lauterbach debugger on EVB

  ***.bin file**

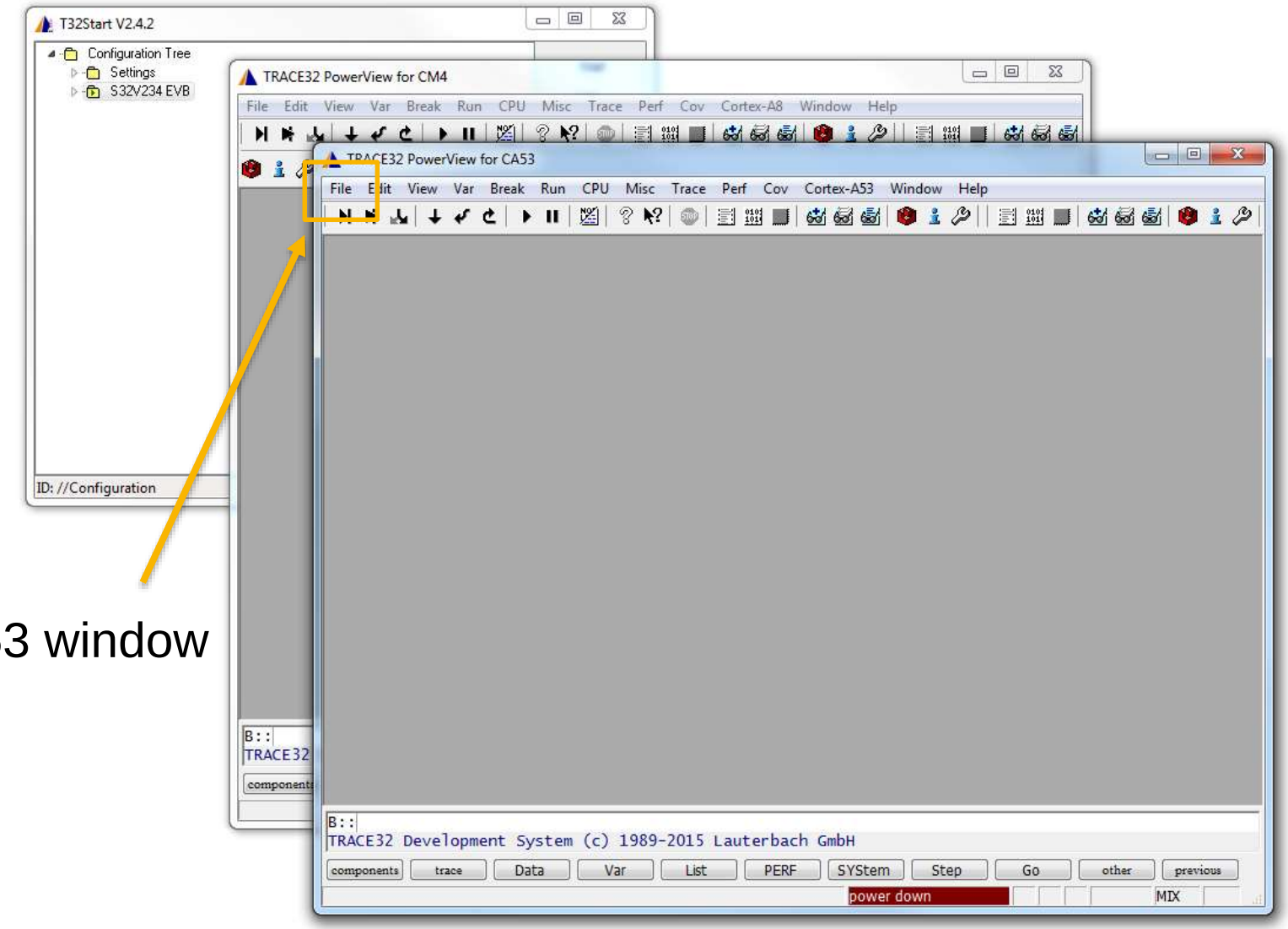
  - Binary file to be loaded on SD card if SD card boot is required

  ***.cmm script**

  - Lauterbach script to be executed in order to load the application via Lauterbach debugger
  - S32V234_SDK_ROOT must be defined pointing to s32v234_sdk directory – automatically set during the install

- s32v234_sdk/os/debug/lauterbach/S32V234.ts2 sets the Lauterbach environment

2.4 Running the demos - Standalone



Open the script in CA53 window

2.4 Running the demos - Linux

- Every demo contains following files after build (build-v234ce-gnu-linux-d):

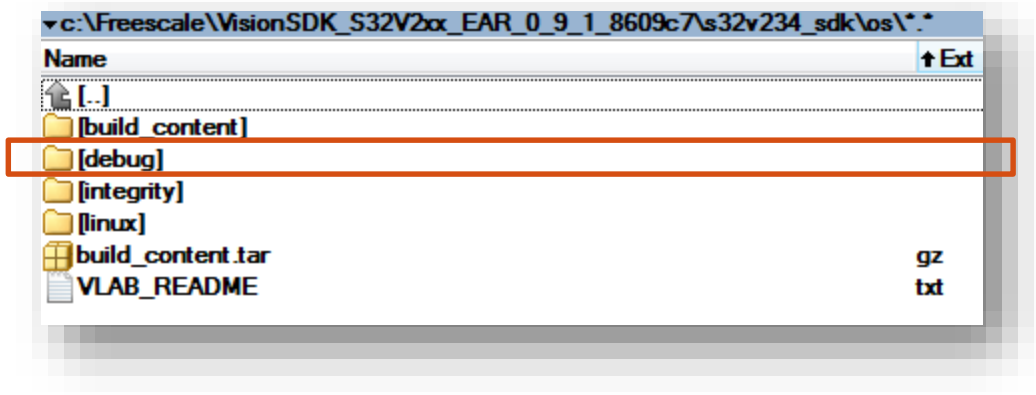
  ***.elf file**

  - Executable to be executed in Linux rootfs on board
  - Shared libraries must be present in rootfs (they are set in provided rootfs)

2.5 Debugging

- Debugging on standalone is the same as running the standalone application via Lauterbach debugger

- Debugging on Linux
  - run s32v234_sdk/os/debug/lauterbach/S32V234-CA53_linux.cmm script to attach to the running Linux instance



SECURE CONNECTIONS
FOR A SMARTER WORLD