AMD Embedded Development Framework (EDF): Product Overview

Tools Product Marketing
Sept 2025

AMD
together we advance_

What is AMD Embedded Development Framework (EDF)?

Robust embedded SW development framework, environment & product, accelerating adaptive SoC adoption

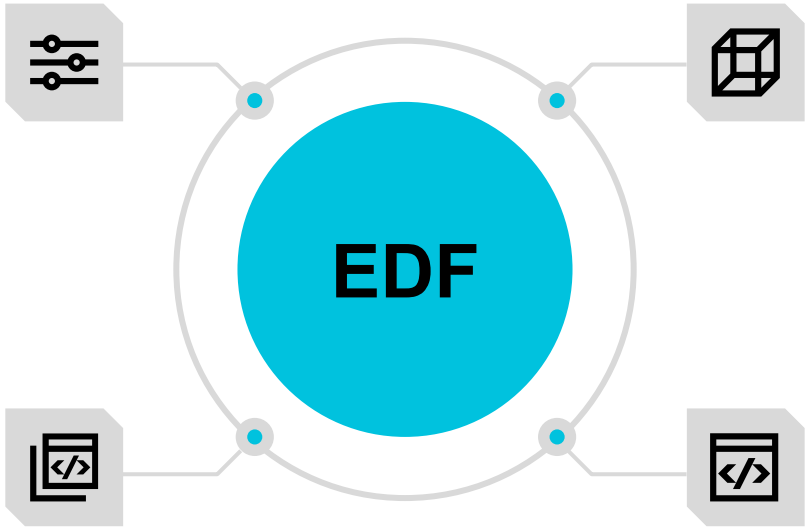
AMD Embedded development framework (EDF) is all of the following

A Methodology

- A way to develop and distribute embedded software components
- A framework for building complex SoC (APU, RPU, PL, AIE etc.)

A Software Development Kit

- An open-source build tool to generate robust Linux® images
- A system or platform configuration



An Out-of-the-box Experience

- Scalable pre-built images to download and quickly run on evaluation boards
- Standard and familiar Yocto Project™ flows for simplified adoption

A Complete Embedded Software Stack

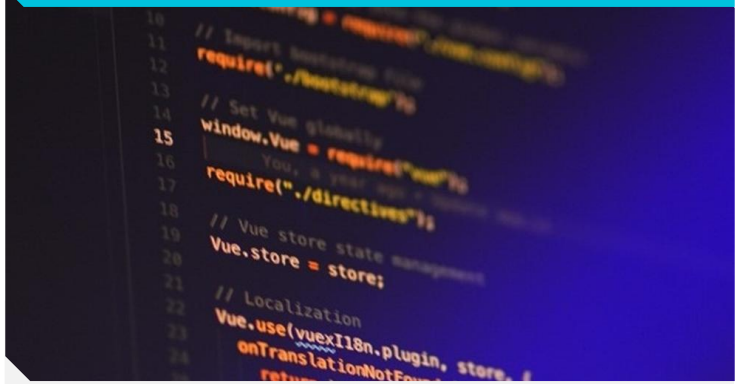
- Includes tool chains, boot firmware, EDF Linux image, example designs and more
- Yocto Project-based OS distribution and an easy handoff to OS vendors

Value of EDF – Fast Time to Market (TTM) & Production Pathway

Discovery

- Uses pre-configured system images
  - Feature rich to showcase the silicon
- Offers easy build flows for complex systems
  - CED based hardware projects
  - Yocto Project™-based software and images
- Deployment on common software stack

Development



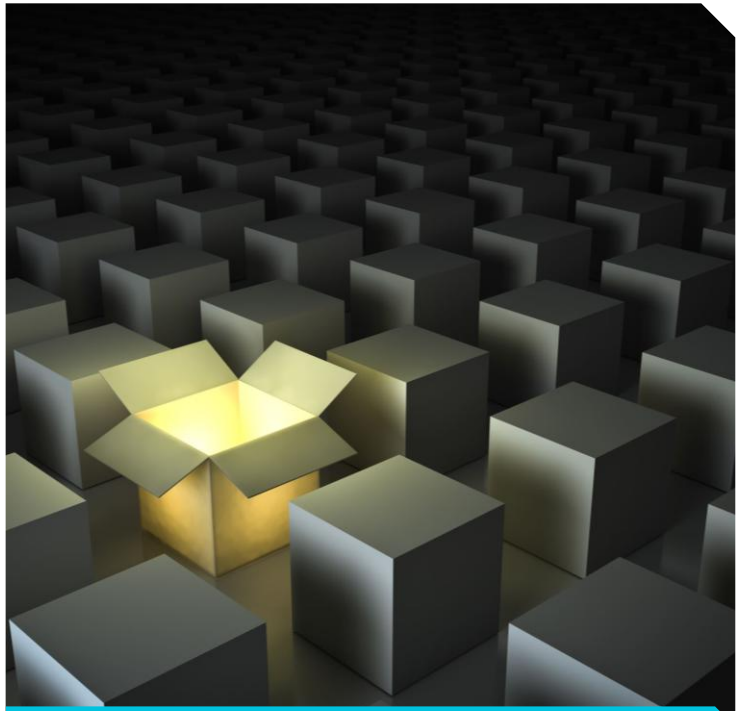
- Allows mix & match from a portfolio of solutions
  - Modular HW/SW reference designs
  - Re-create & extend – with example designs & Yocto Project layers and recipes
- Manages payload from Linux®
- Enables development on host or on target board

Deployment

- Facilitates embedded software flows with pathways to deployment
  - Yocto Project - Self maintain
  - Handoff to an Operating System vendor (non-proprietary tool flows)

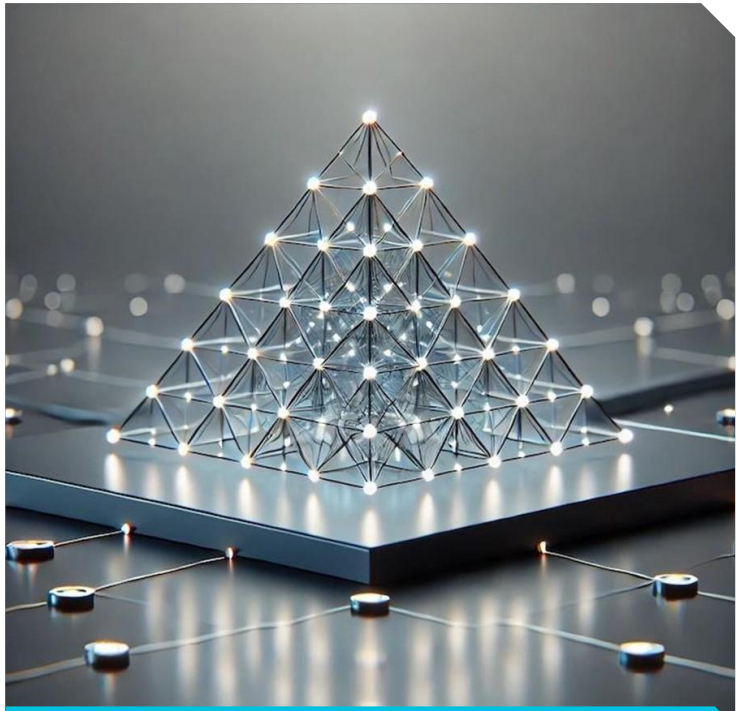
EDF provides system-level solutions and reference flows using industry standards to reduce the development burden

Key Features of EDF



Open Source

Fast Development Path



Complete Solution Stack

Open Source

Open Source Yocto Project – Overview

A toolkit to build custom software stacks for embedded systems

Open-source collaboration for embedded Linux®

Co-maintained with OpenEmbedded Project

Custom Linux distributions for AMD adaptive SoCs

Support for real-time computing constraints

Yocto Project™ vs. Linux® Distribution

Yocto Project is **not** a Linux distribution

- Provides tools, metadata, and processes
- Enables developers to build custom embedded systems

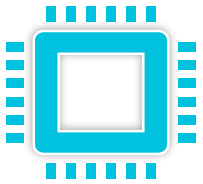
Features

- Modular, standardized, and reproducible build process
- Support for multiple hardware architectures
- Simplified development and maintenance

Customization & Flexibility

- Developers tailor Yocto Project for specific hardware
- Support for unique use cases and system requirements

Host



Linux OS Running on AMD Adaptive SoCs

Key Benefits of the Yocto Project

yocto PROJECT

Custom Distributions
Build OS tailored to specific needs

Broader Ecosystem
Backed by high-tech industry leaders

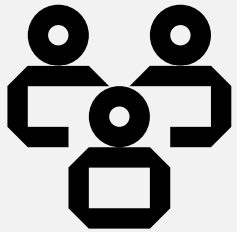
Support for Scalable Board Architecture
Support for x86, Arm®, RISC-V processors, and more

Open-Source Project
Major project under The Linux® Foundation, fully open source

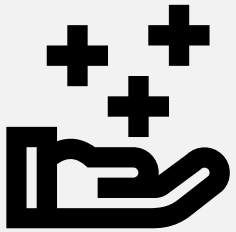
Industry Applications
Used in medical, automotive, and industrial sectors

Standard Toolchains and Libraries
User-friendly tools and libraries for easy integration

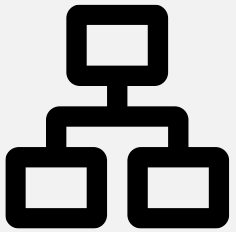
Other Benefits



Full Ecosystem Support:
Access to Open-source Community



Aligned to Yocto Project™ LTS Release:
Long-term Support & Maintenance



Support for Handoff to 3rd Party OS Vendors

Fast Development Path

Different Flows with Different Design Entry Points

Discovery & Evaluation

- *Out-of-the-box (OOB) Experience*
- *Pre-built images*
- *Advanced examples*
- *Reference Designs*

Deployed via pre-built images, Package feeds and docker hub

Pre-built binary image 

Application Development & Deployment

Software Development

- *Software Applications*
- *Drivers*

Hardware Development

- *Compatible PL design using AMD Vivado™ Design Suite*
- *AMD Vitis™ IDE flows*

Deployment (SW & HW)

Test on hardware

Pre-built binary image 

Application packages and container images 

OS Integration & Development

OS Development

- *OS customizations (Yocto Project™ flows)*
- *Software domain configuration*

System Integration

- *Custom hardware import*

New boot firmware / OS image 

Optional – Re-use HW design

Custom Hardware Development

(Traditional starting point)

Hardware Development

- *Custom PL design (not compatible with pre-built images)*
- *Custom boards*
- *Custom isolation and protection*

Software Development and System Integration

- *Images for custom targets*
- *EDF stack can be used if desired*

New boot firmware / OS image 

New HW design artifacts 

Focus on the custom flows – *Re-use the pre-built items, compile only when essential*

Discovery & Evaluation

Discovery & Evaluation

- *Out-of-the-Box (OOB) experience*
- *Pre-built images*
- *Advanced examples*
- *Reference Designs*

Deployed via pre-built images, Package feeds and docker hub

Pre-built binary Image

Out-of-the-box Experience

Unpack the board

Setup the board: Connect/Update AMD Versal™ System Controller

Download pre-built image, flash OSPI, SD card & boot

Load prebuilt PL image

Explore prebuilt application running on board

Application Development & Deployment

Application Development & Deployment

Software Development

- *Software Applications*
- *Drivers*

Hardware Development

- *Compatible PL design using Vivado™ Design Suite*
- *AMD Vitis™ IDE flows*

Deployment (SW & HW)

Test on hardware

Pre-built binary Image

Application packages and container images

Vivado & Vitis Flows: Known Experience

	Software Development Flow	Hardware Development Flow
	Unpack and set up the board	Unpack and setup the board
	Download pre-built image, flash OSPI, SD card & boot	Download pre-built image, flash OSPI, SD card & boot
	Develop the software application on running board	Standard Vitis / Vivado flows to modify or extend the hardware or kernels
	Download an example design to the running board	Verify the compatibility of PL image using pr_verify
	Deploy to the board or re-build the image on your machine using common tools	Download and deploy custom PL PDI, HW kernels, SW app to running board

OS Integration & Development Using Yocto Build Flow

OS Integration & Development

OS Development

- *OS customizations (Yocto Project™ flows)*
- *Software domain configuration*

System Integration

- *Custom hardware import*

New boot firmware / OS image 

Optional – Re-use HW design

Yocto Build Flow

Start with Yocto Project-based EDF Linux® BSP (board-specific)

Integrate different payloads into the build

Customize the OS build

Build and package the disk image

Boot and explore on running on board

Custom Hardware Development

Custom Hardware Development

Hardware Development

- *Custom PL design (not compatible with pre-built images)*
- *Custom boards*
- *Custom isolation and protection*

Software Development and System Integration

- *Images for custom targets*
- *EDF stack can be used if desired*

New boot firmware / OS image

New HW design artifacts

Custom Board Flow

Create a custom hardware using AMD Vivado™ Design Suite

Generate the boot.pdi and pl.pdi

Export XSA from Vivado Design Suite

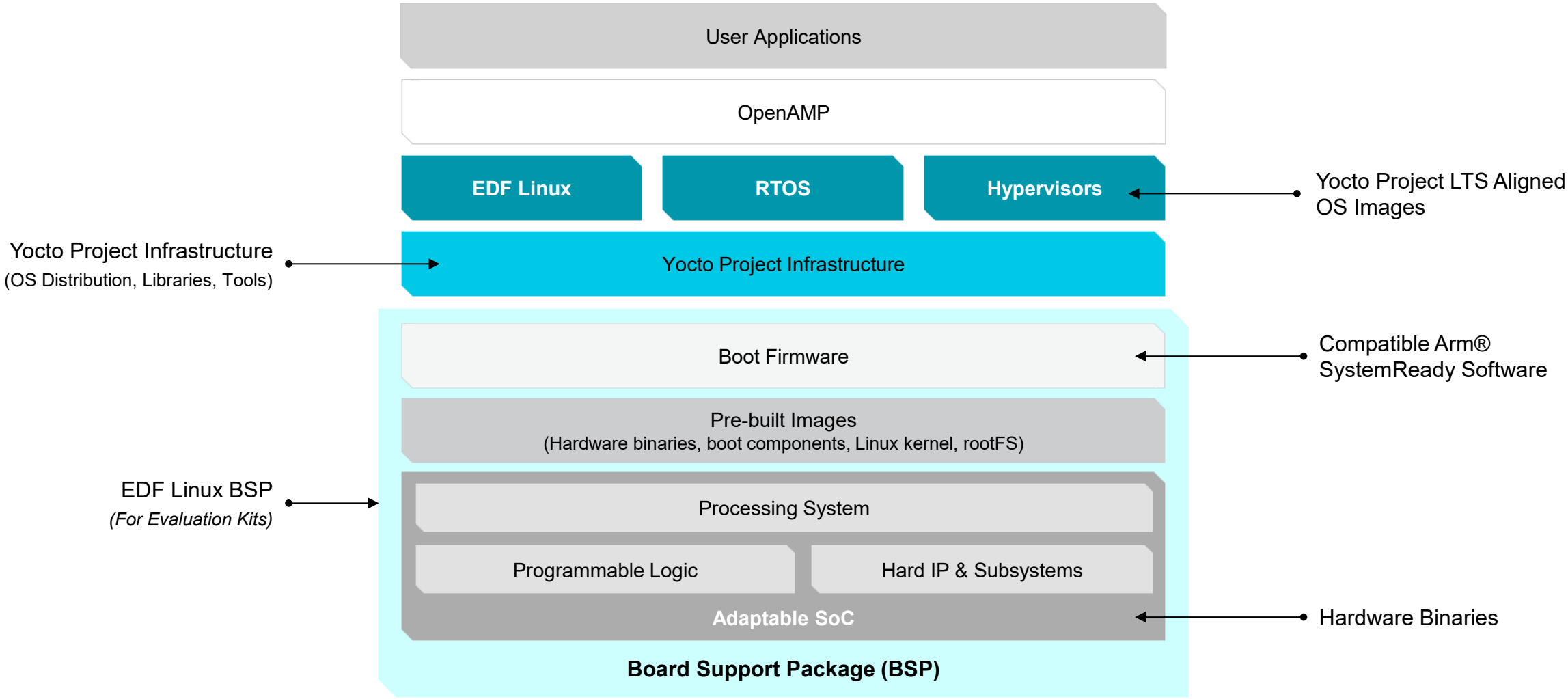
Handoff the XSA and integrate it into software flows using SHEL

Run the Yocto Project build and generate OS image

Deploy to the custom board

Complete Solution Stack

EDF Software Stack



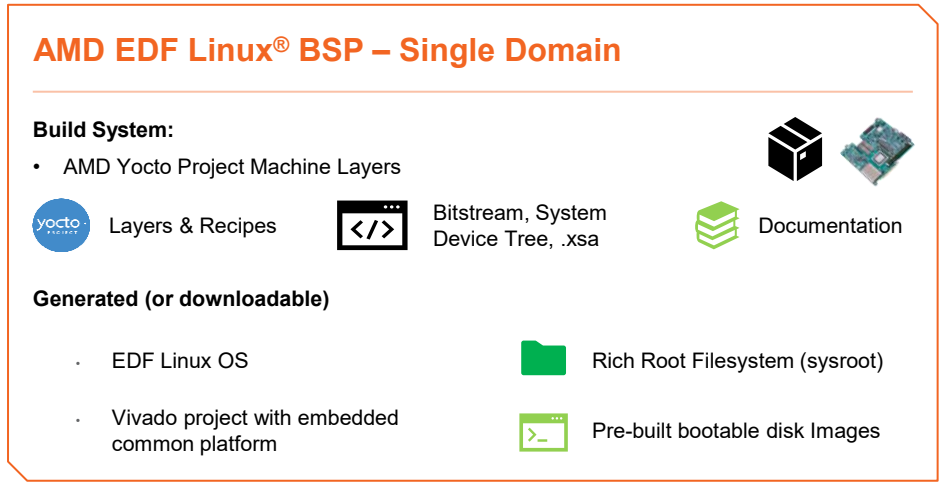
AMD EDF Linux BSP and Pre-built Disk Images

Provides you everything needed to build a running system on AMD adaptive SoCs

AMD EDF Linux® disk image created using the board support package (BSP) –

- BSP contains embedded software configuration (Yocto Project™-based) and hardware design description (AMD Vivado™ Design Suite based)
- AMD Vivado™ Design Suite project - Downloadable as a Tcl file
- Pre-built Vivado artifacts - Downloadable & used by Yocto Project build
- Prebuilt EDF Linux disk image contains:
  - EDF boot firmware and Linux utilities
  - Compatible Arm® SystemReady software boot flow
  - Rich Linux OS: utilities, libraries, compilers
  - FPGA Manager / DFX Manager Linux utilities for PL firmware management
  - AMD Vitis™ Unified IDE Support: Run the Vitis kernel flow on the pre-built image*
  - Software and hardware reference designs via package feed for download and run
  - PL payload - BSP base platform: RAM, GPIO etc.

AMD EDF Linux® BSP – Single Domain



* Support is currently is limited to AMD Versal™ AI Edge Series Gen 2 and Versal Prime Series Gen 2 devices, as of October 2025. AMD expects future products will support AMD Vitis kernel flow in the EDF environment.

AMD EDF Platform BSP – Multi-Domain

AMD EDF Platform BSP disk image built on top of the AMD EDF Linux BSP –

- Unified binary disk image per device family
- Application processor (Arm® Cortex®-A) support
  - EDF Linux® OS
  - Xen® Hypervisor
  - Canonical® Ubuntu® via separate download
- Real time processor (Cortex-R) support
  - Zephyr® RTOS via parallel boot or via OpenAMP
- Additional pre-defined configurations
- Default boot mode – Multi-stage with deferred PL load
- Support for reference designs and platforms

EDF Platform BSP – Multi Domain

- Unified binary disk images per family
- Reference platforms with isolation and domains
- Reference embedded software payloads
- Modular TRD / ARD & OTA Updates
- Segmented Configuration
- Yocto Project™ best practices*

Linux OS, Zephyr, Xen and Yocto

AMD EDF Linux® BSP – Single Domain

Build System:

- AMD Yocto Project Machine Layers

Layers & Recipes

Bitstream, System Device Tree, .xsa

Documentation

Generated (or downloadable)

- EDF Linux OS
- Vivado™ project with Embedded Common Platform

Rich Root Filesystem (sysroot)

Pre-built Bootable disk Images

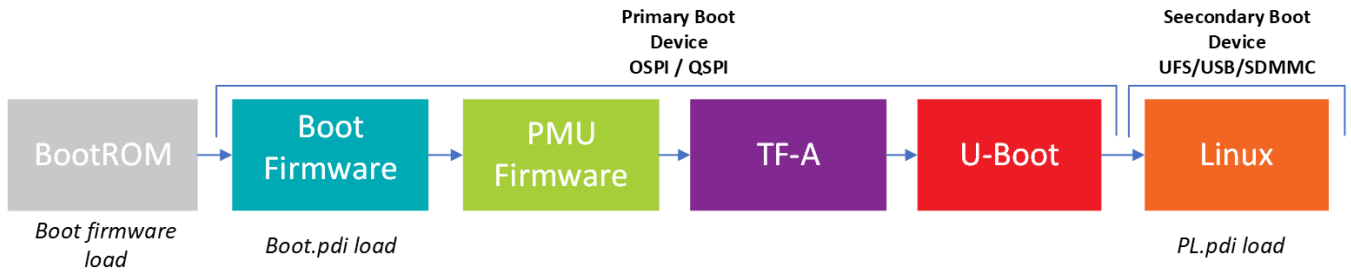
EDF Boot Flows Support

All boot and configuration options are supported with EDF tools

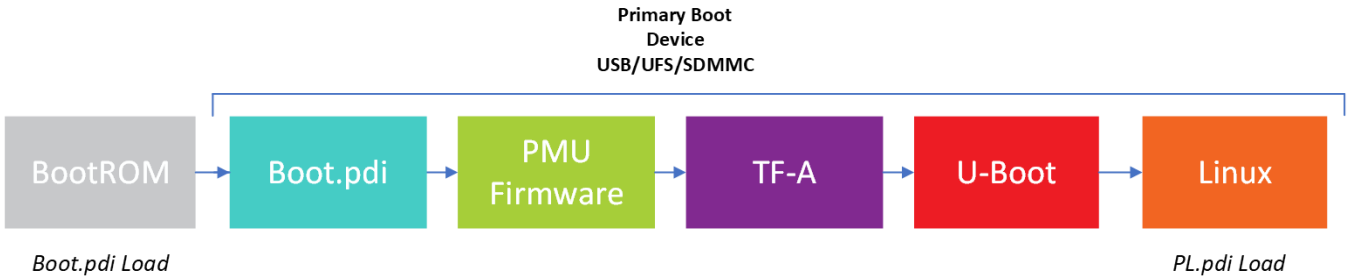
1. Single stage, multi-stage
2. All boot devices – JTAG, QSPI, OSPI, SD card, eMMC, UFS
3. Segmented configuration, monolithic configuration
4. Advanced – PCIe®, GbE, Tandem Boot

Default boot architecture for EDF Linux ®, BSP

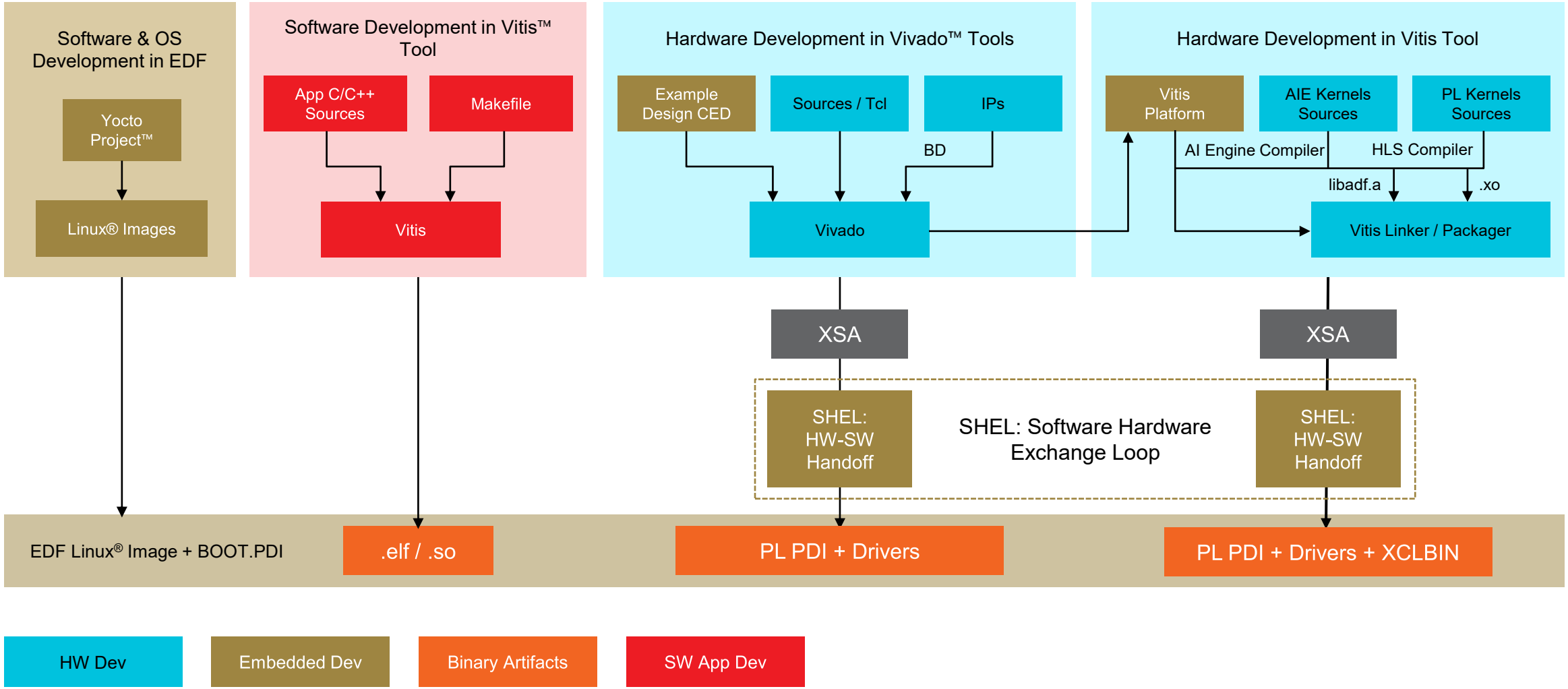
For AMD Versal™ AI Edge Series Gen 2 and beyond: Multistage boot with deferred PL load



For AMD Versal™ Series Gen 1 and previous architectures (including AMD Zynq™ devices): SD card boot with deferred PL load



EDF Integrated into Different Flows



Software Application Development & Deployment

Development

EDF provides two options to develop software application:

On Target Development

Convenient due to common compilers, tools, and workflow is used

Development Using SDK

SDK for cross development on x86 processor – can speed up the build but fixed

Deployment

EDF supports standard Linux® / Yocto Project™ for deployment and integration

Deployment via a container for upload to a container registry or **direct transfer** to a running system

- Creation of a container image with all dependencies: Drivers, packages, application software, and firmware (PDI)
- Useful for dependency separation and pushing to running systems as a semi-standalone package
- For direct transfer: Download and install the SDK from amd.com, compile and transfer to running target
- Integration into the main Yocto Project build – Additional layers / recipes appended to the build with installation to the rootfs

Use of a package feed – A convenient way of installing the additional packages into the user's root file system

- EDF has package manager and AMD has its own packege feed
- Creation of a package of dependencies, deployed via a package feed for install (pull) from user space on the target
- Access to kernel source packages, compilers and tool chains
- Pathway for OTA updates and distribution flow for TRD, firmware, application examples etc.

Custom Hardware Development – AMD Vivado CED Flow

Vivado™ **Configurable Example Design (CED)** – provides a common embedded platform

- Accessed using Vivado -> New Project -> from example
- Enables embedded software image and compatible custom PL images
- Acts as a starting point for BSP, board-based TRD, platforms
  - Used by AMD Vitis™ tool, IP, AIE, ISP and Embedded Software
  - Vivado Board Support Automation (VBS) + TCL + GUI
  - Mapped to eval boards: Implements necessary board specifications
- Alternative payloads – hierarchical & modular
  - Choose PL payload – Default, Vitis Extensible Platform, EDF BSP, or other payload

Board based CED

- PS IP and Board Interfaces
- Vivado Board File (VBS) automation

Common Base (PS Setting + NOC)

Evaluation Board Features

\+ Select PL Payload

Minimal PL (DEFAULT)

MMIP Platform : PL + Hard IP

Vitis extensible : PL + Hard IP

BSP base platform

Other

Other

Vivado CED Flow

Create a New Vivado Example Design

↓

Modify or Extend PL Payload

↓

Generate Output Products

↓

Validate Compatibility (pr_verify)

↓

Generate XSA

↓

Hardware Handoff to Software

↓

Deploy on Custom Board

SHEL – Software Hardware Exchange Loop: Introduction

It's more than just hardware to software translation

Enables developers to mix and match tools from different vendors and versions

- Defines a way for system, hardware, and software engineers to communicate about hard, firm and soft components of the system and their interactions
- Decouples hardware and software and makes sure all the different components like hardware, firmware, software, or entire operating systems stay in sync

Access to the external documentation and expertise

- Part of AMD Embedded Development Framework (EDF), Non-proprietary
- Connection to open-source tools with full ecosystem support which makes it easier for customers to carry experience

Allows integration of AMD tools into customer's existing flow

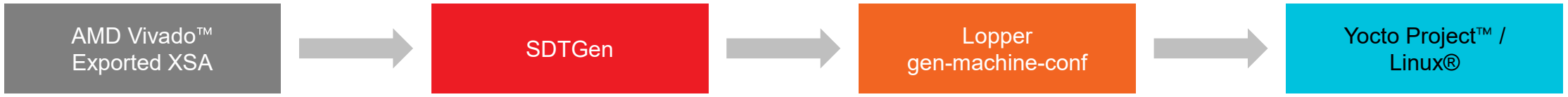
- SHEL takes AMD proprietary XSA and translates it into open-source hardware description - System Device Tree (SDT)
- Allows customers to pick and choose which bits of our flow they want to integrate into their existing flow/processes

Safe and secure software stacks

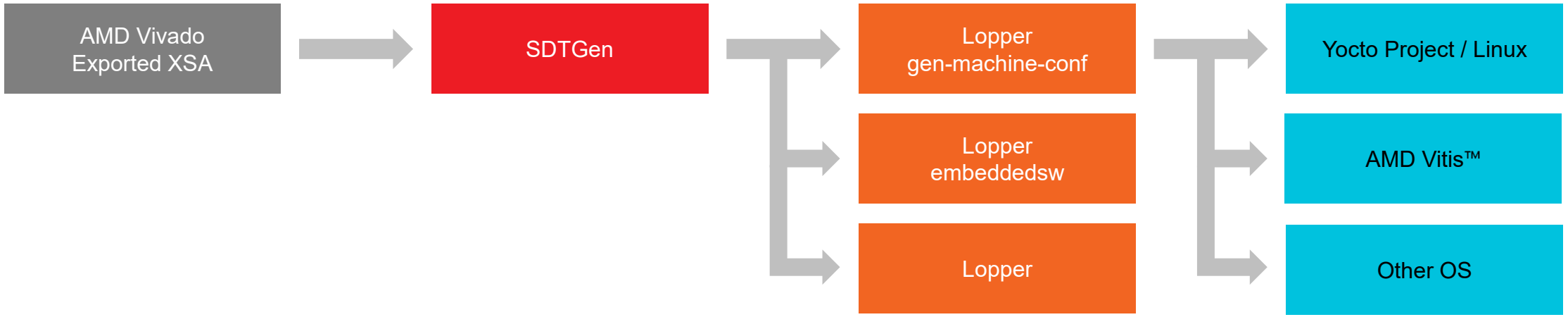
- Allows usage of multi-domain systems to simplify safety and security
- Industry standards like Yocto Project™, Xen, OpenAMP, and Zephyr are enabled by SHEL

SHEL – Software Hardware Exchange Loop: How It Works?

SHEL Flow – Single Domain –Linux® build, and configuration data generated for further development using Yocto Project



SHEL Flow – Multi Domain – Linux build, device tree, and configuration data generated for Yocto Linux development, embedded software development, and OS development



AMD example designs use the SHEL flow

AMD EDF Terminology

EDF	Embedded Development Framework
BSP	Board Support Package
AMD EDF Linux BSP	A single domain Linux® pre-built image containing everything needed to build a running system for a specific board
AMD EDF Platform BSP	A multi-domain pre-built image supporting Linux + RTOS + Hypervisors + other pre-defined configurations
Yocto Project Build	A process of creating a custom Linux-based OS image for embedded devices using the Yocto Project™
Yocto Recipe and Layers	Collection of files containing the instructions and related metadata used by the build system (BitBake) to fetch, configure, compile, and install a specific software package
CED	Configurable Example Design in AMD Vivado™ Design Suite
XSA	Xilinx Support Archive – Hardware design exported from Vivado Design Suite
BIT/PDI	Binary file representing hardware design

Boot Firmware	OSPI / QSPI based initial boot software
SHEL	Software Hardware Exchange Loop – A process to facilitate the exchange of hardware and software configuration data among various open-source and proprietary tools
SDT – System Device Tree	Provides a holistic view of the entire system's hardware configuration
Device Tree	Used only by the Linux kernel to describe hardware components
SDTGen	A tool that translates proprietary output from the Vivado Design Suite into a stable, open-source, text-based format
Lopper	A tool that processes and transforms device tree data, facilitating seamless integration and automation across different software domains
OSV	Operating System Vendor
Zephyr	Open-source real-time operating system (RTOS)
Hypervisor	A software that allows to run multiple virtual machines on a single physical machine

AMD EDF – Getting Started Tutorials and Documentation

Embedded Development Framework

AMD Embedded Development Framework

- Development Flows - How it all works
- Getting Started with EDF
- Discovery & Evaluation
- Application Development & Deployment
- OS integration & Development
- Custom Hardware Development

Customer Training Course on Yocto Project™

- Embedded Linux Development Using Yocto Project

DISCLAIMER AND ATTRIBUTIONS

DISCLAIMER: The information contained herein is for informational purposes only and is subject to change without notice. While every precaution has been taken in the preparation of this document, it may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this information. Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of this document, and assumes no liability of any kind, including the implied warranties of noninfringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by this document. Terms and limitations applicable to the purchase or use of AMD products are as set forth in a signed agreement between the parties or in AMD's Standard Terms and Conditions of Sale. GD-18u.


AMD