EDK Overview

XILINX®

Objectives

After completing this module, you will be able to:

- Describe the embedded systems development flow
- Describe the components in the hardware design
- Specify ways to create a hardware design
- Identify the tools included in the Embedded Development Kit (EDK)
- List the supported operating systems

Outline

- **Introduction**
- EDK
  - Overview of EDK
  - Embedded Development Design Flow
  - Embedded Project Management
- Supported Platforms

Embedded System

- An embedded system is nearly any computing system (other than a general-purpose computer) with the following characteristics:
  - Single function
    - Typically designed to perform a predefined function
  - Tightly constrained
    - Tuned for low cost
    - Single-to-fewer component based
    - Performs functions fast enough
    - Consumes minimum power
  - Reactive and real-time
    - Must continually monitor the desired environment and react to changes
  - Hardware and software coexistence

Embedded Systems

- Examples:
  - Mobile phone systems
    - Customer handsets and base stations
  - Automotive applications
    - Braking systems, traction control, airbag release systems, and cruise-control applications
  - Aerospace applications
    - Flight-control systems, engine controllers, auto-piloting systems, and passenger in-flight entertainment systems
  - Defense systems
    - Radar systems, fighter aircraft fire-control systems, radio systems, and missile guidance systems

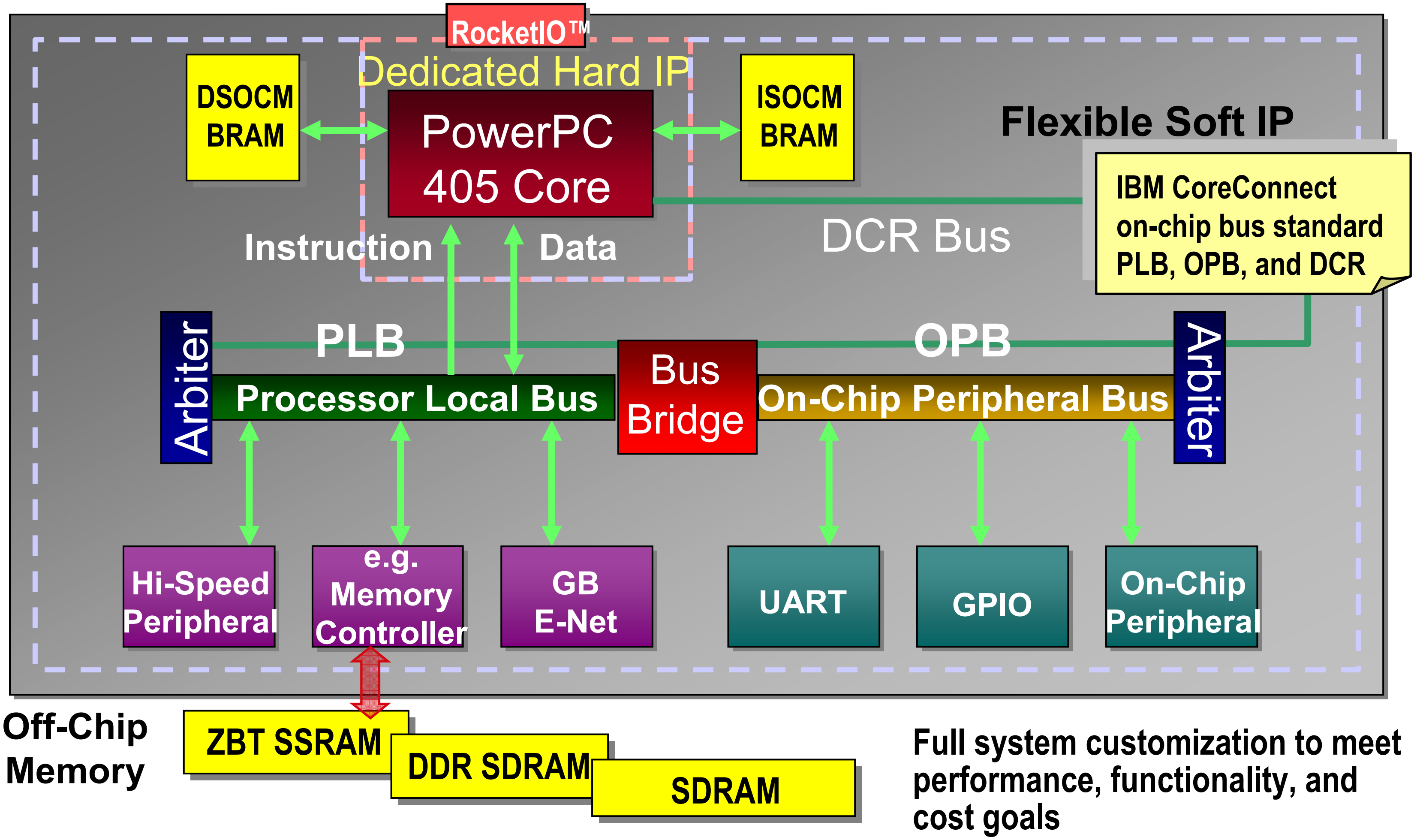
Current Technologies

- Microcontroller-based systems
- DSP processor-based systems
- ASIC technology
- FPGA technology

Embedded Design in an FPGA

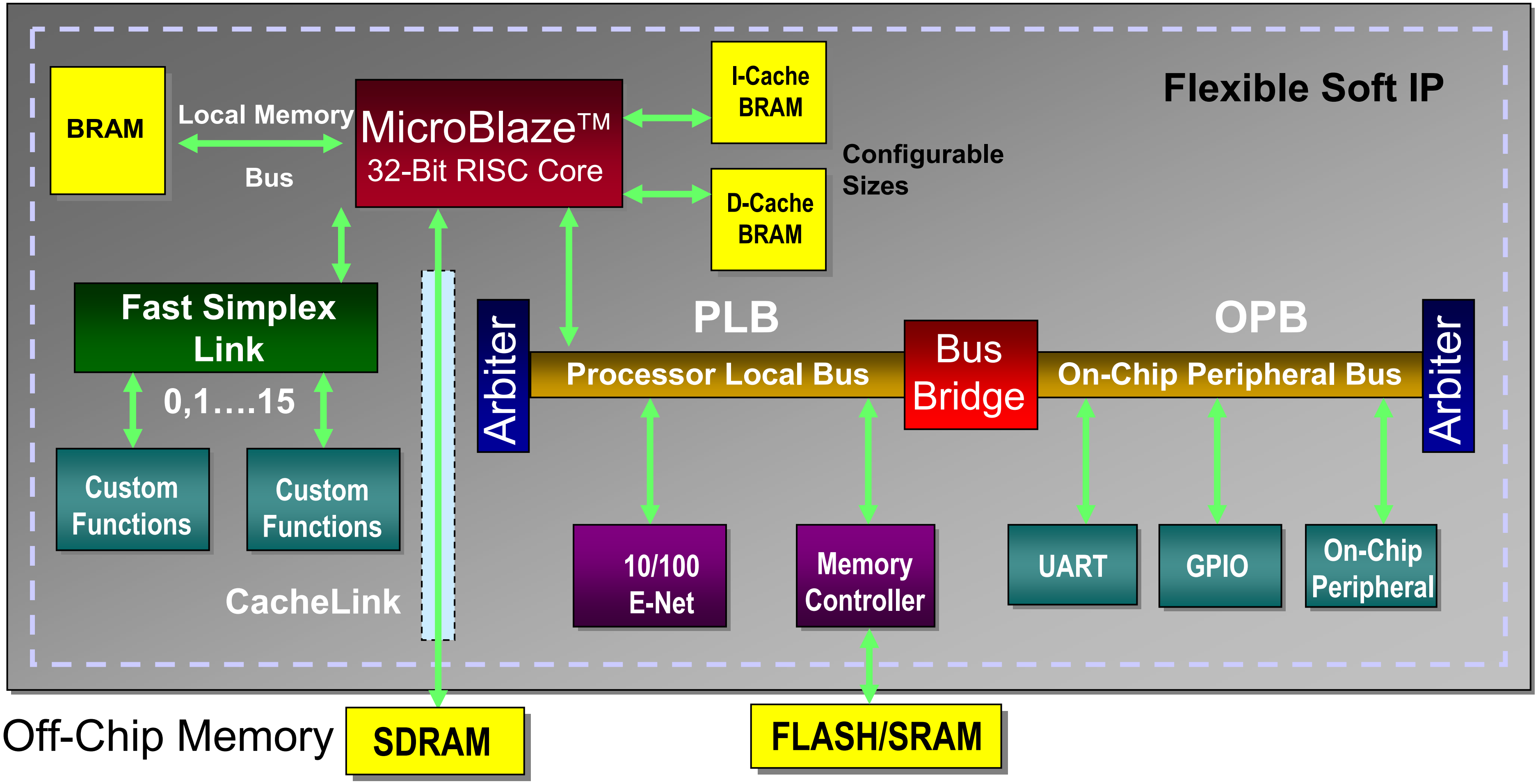
- Embedded design in an FPGA consists of the following:
  - Develop FPGA hardware design
  - Generate drivers and libraries
  - Create the software application
    - Software routines
    - Interrupt service routines (optional)
    - Operating System (OS) or Real Time Operating System (RTOS) (optional)

PowerPC Processor-Based Embedded Design



Full system customization to meet performance, functionality, and cost goals

MicroBlaze Processor-Based Embedded Design



This is a v7.0 architecture. Versions 6.0 or earlier do not support PLB bus off the processor. Instead they have OPB bus

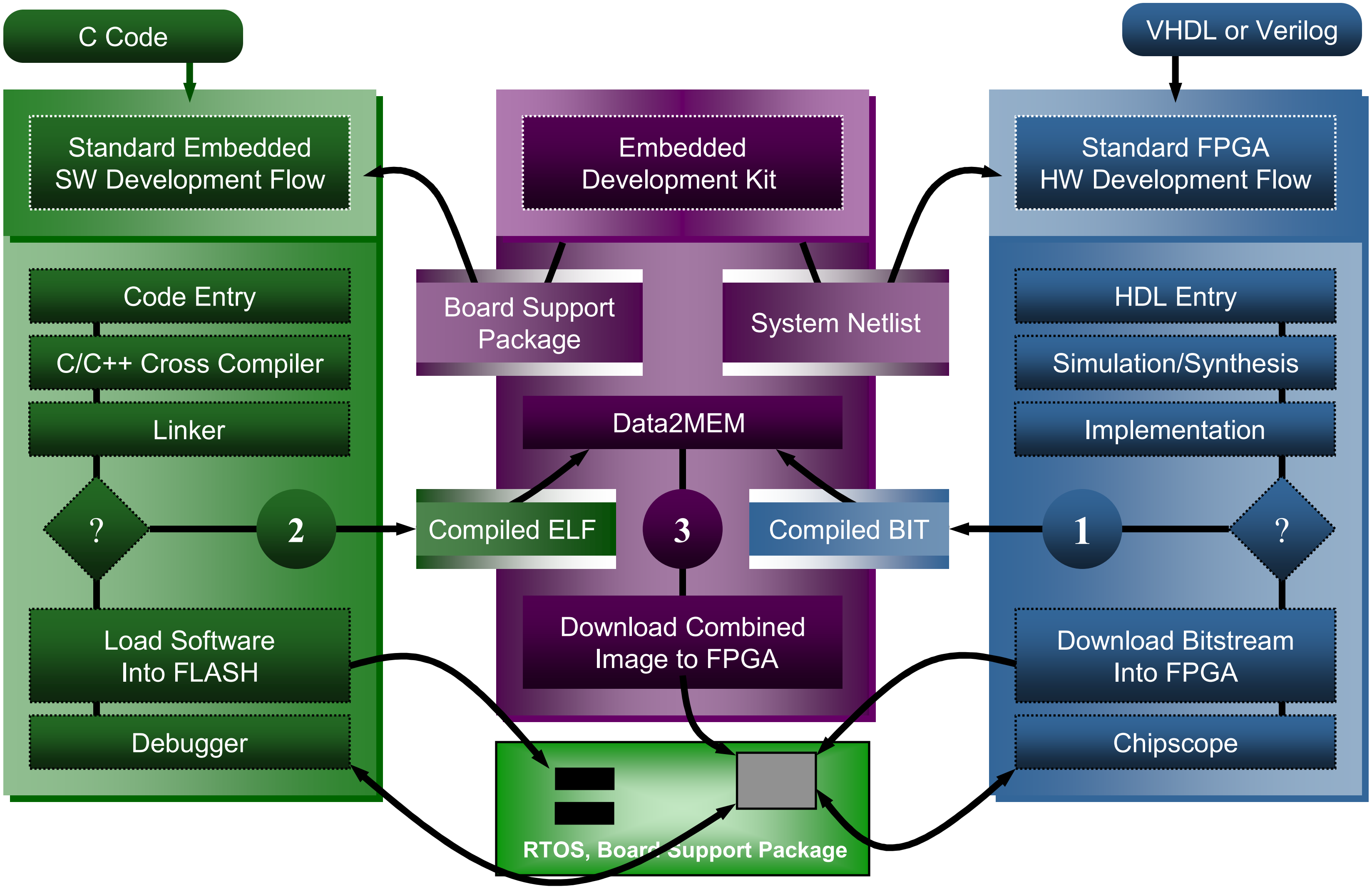
Outline

- Introduction
- **EDK**
  - **Overview of EDK**
  - Embedded Development Design Flow
  - Embedded Project Management
- Supported Platforms

Embedded Development Kit

- What is Embedded Development Kit (EDK)?
  - The Embedded Development Kit is the Xilinx software suite for designing complete embedded programmable systems
  - The kit includes all the tools, documentation, and IP that you require for designing systems with embedded IBM PowerPC™ hard processor cores, and/or Xilinx MicroBlaze™ soft processor cores
  - It enables the integration of both hardware and software components of an embedded system

Embedded Development
Tool Flow Overview



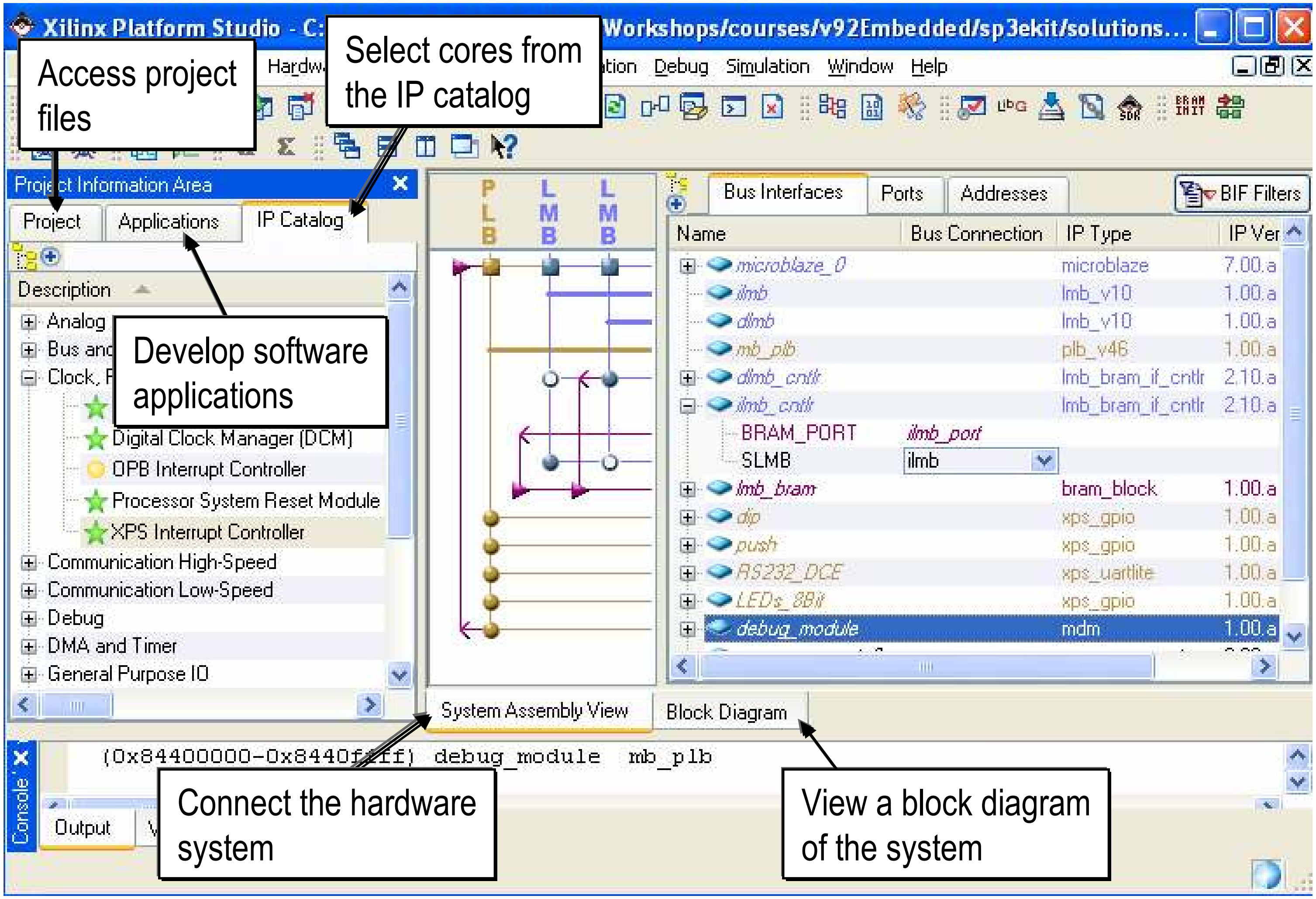
Embedded System Tools

- Develop the embedded hardware
  - Quickly create a system targeting a board using **Base System Builder Wizard**
  - Extend the hardware system
    - Add peripherals from the **IP Catalog**
    - Create and add a custom peripheral using the **Create/Import Peripherals Wizard**
    - Insert ChipScope™ Pro cores into the system using the **Debug Configuration Wizard**
  - Generate HDL netlists using **PlatGen**
  - Perform an HDL simulation using an HDL simulator
    - Generate simulation models using **SimGen**
- Develop the embedded software
  - Generate libraries and drivers with **LibGen**
  - Create and debug the software ap in **XPS** or the **Software Development Kit (SDK)**
    - Compile using the **GNU C/C++ compiler (gcc)**
    - Connect to the target using **Xilinx Microprocessor Debug (XMD)**
    - Debug using the **GNU debugger (gdb)**
- Operate in hardware
  - Generate the bitstream and configure the FPGA
    - The **bitstream initializer (BitInit)** will update FPGA instruction memory with the executable
  - Initialize external flash memory
    - Write to external flash using the **Flash Writer utility**
    - Generate an external compact flash configuration file using the **System ACE File generator (GenACE)**

Embedded System Tools

- Libraries
  - lwIP Library – A third party network library ported to Xilinx embedded processors
  - LibXil MFS – A memory file system
  - LibXil FATfs – A FAT file system
  - LibXil Flash – A flash memory support
  - Standard C libraries (libc, libm)
- Operating System Board Support Packages (BSPs)
  - Standalone operating system
  - Xilinx MicroKernel (XMK)

Xilinx Platform Studio (XPS)



Name	Bus Connection	IP Type	IP Ver
microblaze_0		microblaze	7.00.a
ilmb		lmb_v10	1.00.a
dlmb		lmb_v10	1.00.a
mb_plb		plb_v46	1.00.a
dlmb_cntlr		lmb_bram_if_cntlr	2.10.a
ilmb_cntlr		lmb_bram_if_cntlr	2.10.a
BRAM_PORT	ilmb_port		
SLMB	ilmb		
lmb_bram		bram_block	1.00.a
dip		xps_gpio	1.00.a
push		xps_gpio	1.00.a
RS232_DCE		xps_uartlite	1.00.a
LEDs_8Bit		xps_gpio	1.00.a
debug_module		mdm	1.00.a

```
(0x84400000-0x8440ffff) debug_module   mb_plb
```

Access project files

Select cores from the IP catalog

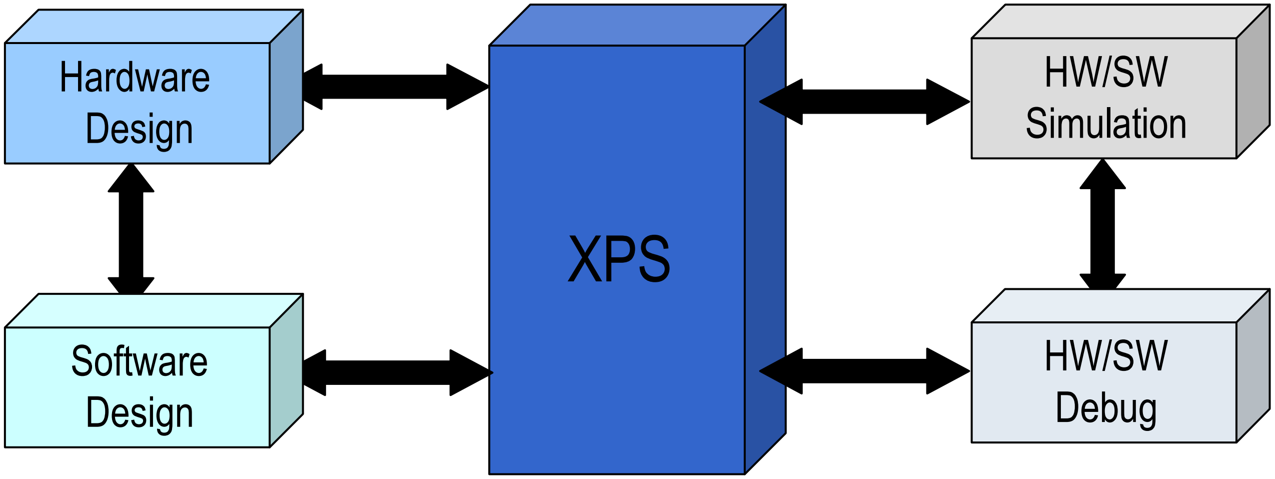
Develop software applications

Connect the hardware system

View a block diagram of the system

XPS Functions

- Project management
  - Microprocessor Hardware Specification (MHS) or Microprocessor Software Specification (MSS) file
  - Xilinx Microprocessor Project (XMP) file
- Software application management
- Platform management
  - Tool flow settings
  - Software platform settings
  - Tool invocation
  - Debug and simulation



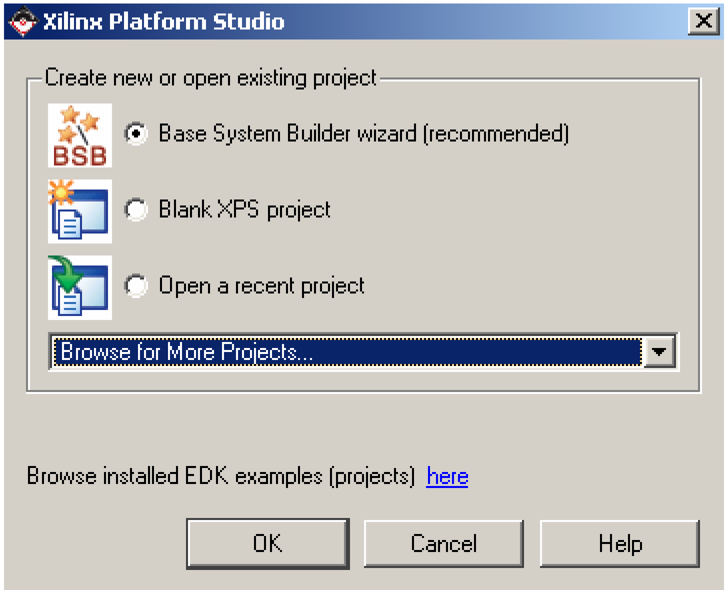
Outline

- Introduction
- **EDK**
  - Overview of EDK
  - **Embedded Development Design Flow**
  - Embedded Project Management
- Supported Platforms

Create/Open a Project

- Create a new project
  - Using **File → New Project**
    - Select Base System Builder option
    - Select Blank XPS Project option
- Open an existing project
  - Using **File → Open Project**
    - Browse to a pre-created xmp file
  - Using **File → New Project**

    Select Open a Recent Project option and selecting a project



Xilinx Platform Studio

Create new or open existing project

● Base System Builder wizard (recommended)

○ Blank XPS project

○ Open a recent project

Browse for More Projects...

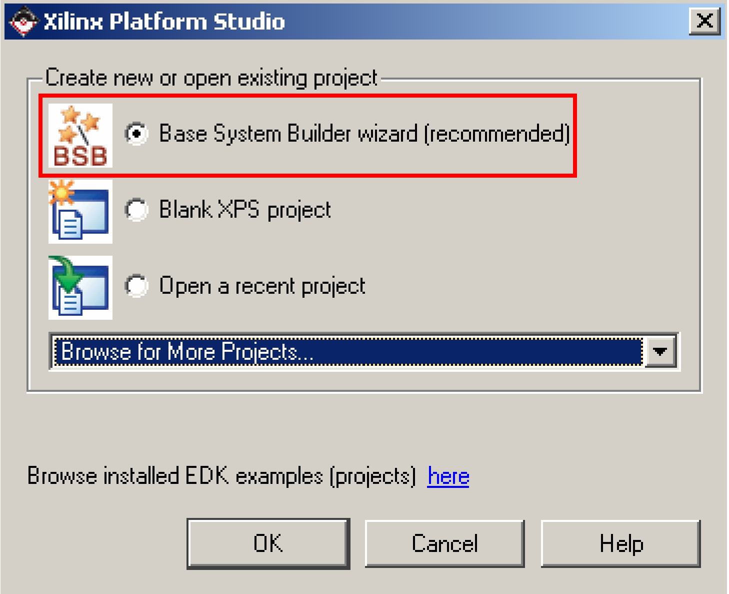
Browse installed EDK examples (projects) here

- Project information is saved in the Xilinx Microprocessor Project (XMP) file

Create a Project

Automatically with Base System Builder

- Select a target board
- Select a processor
- Configure the processor
- Select and configure I/O interfaces
- Add internal peripherals
- Generate the system software and the linker script
- Generate the Design



Xilinx Platform Studio

Create new or open existing project

● Base System Builder wizard (recommended)

○ Blank XPS project

○ Open a recent project

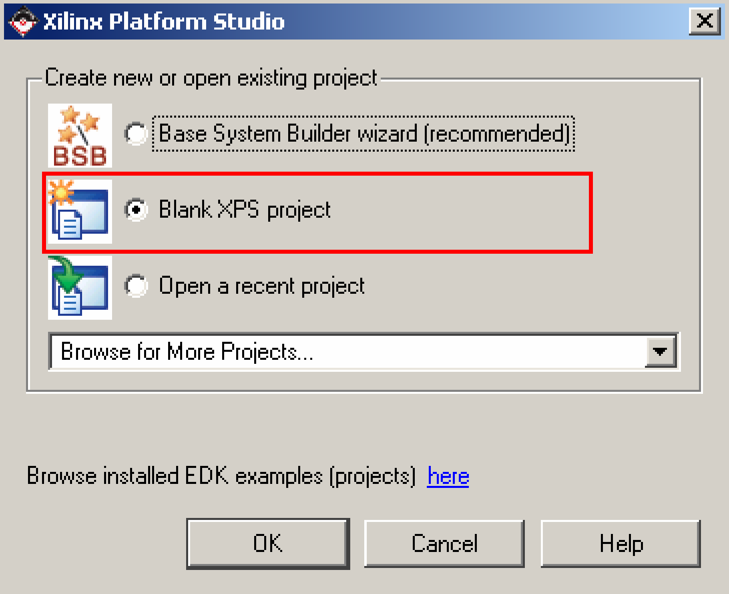
Browse for More Projects...

Browse installed EDK examples (projects) here

- **Recommended for initial project creation**

Create a Project
From Scratch

- Identify a New XPS Project location
  - Select target FPGA, and optionally import an MHS file and/or user repository directory
- Use IP Catalog, add processor(s) & peripheral(s)
- Configure the processor(s) and peripheral(s)
- Create UCF file
- Specify Software Configuration for the hardware components
- Develop application software
- Generate Bitstream
- Download Bitstream and Execute


Xilinx Platform Studio

Create new or open existing project

○ Base System Builder wizard (recommended)

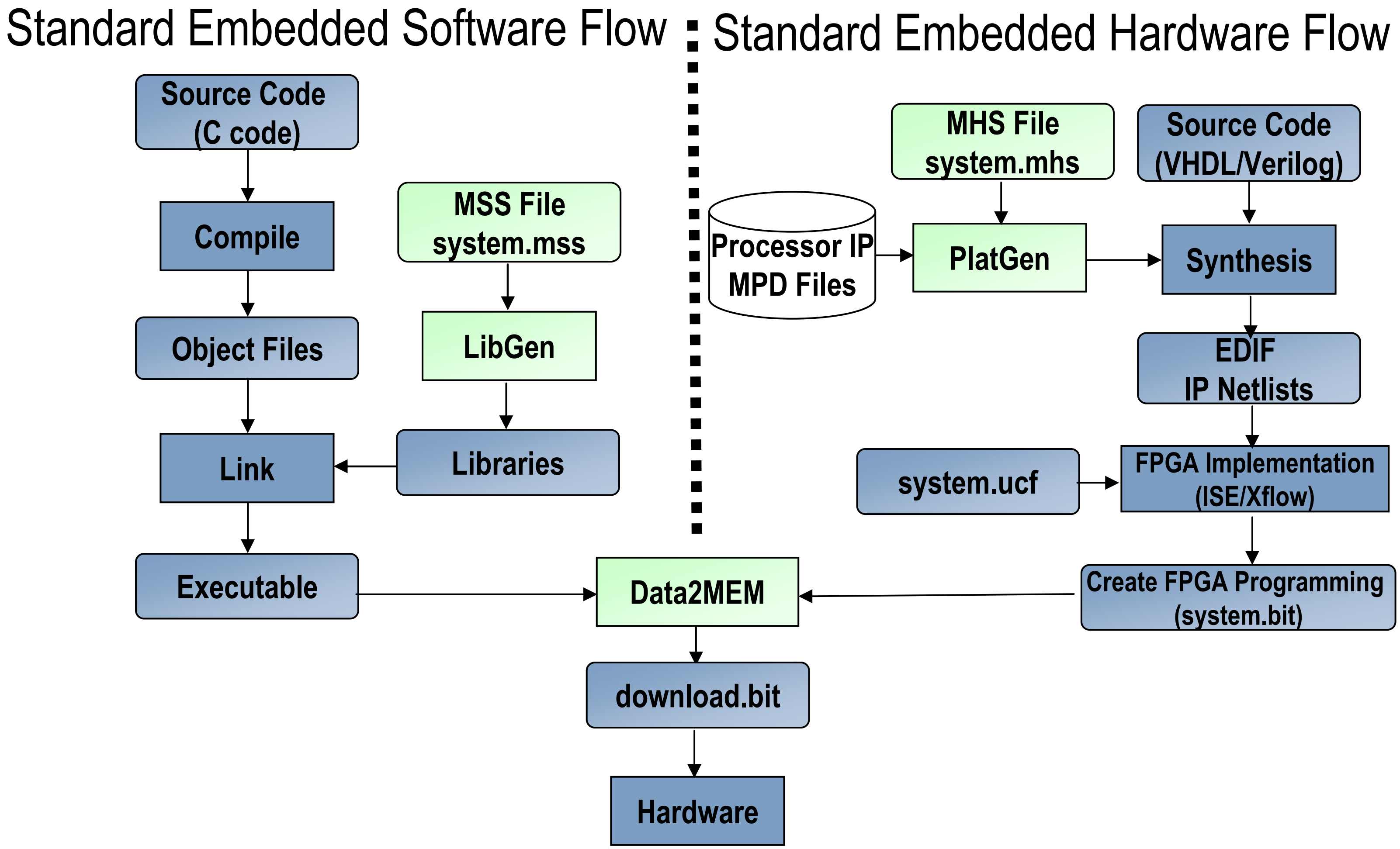
● Blank XPS project

○ Open a recent project

Browse for More Projects...

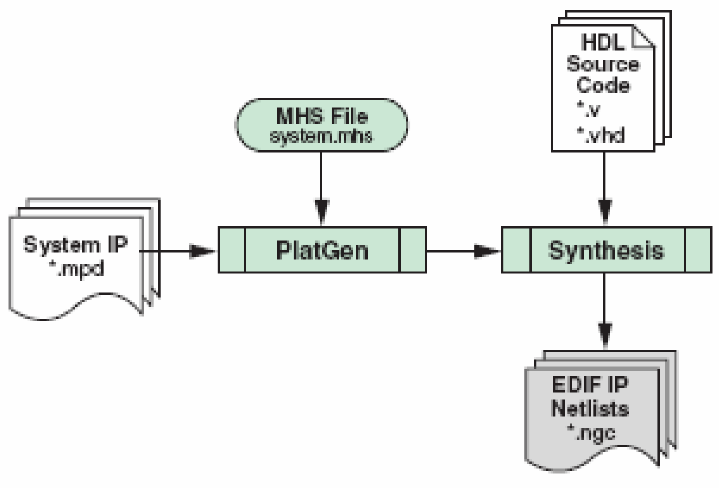
Browse installed EDK examples (projects) here

Detailed EDK Design Flow



Hardware Creation Flow

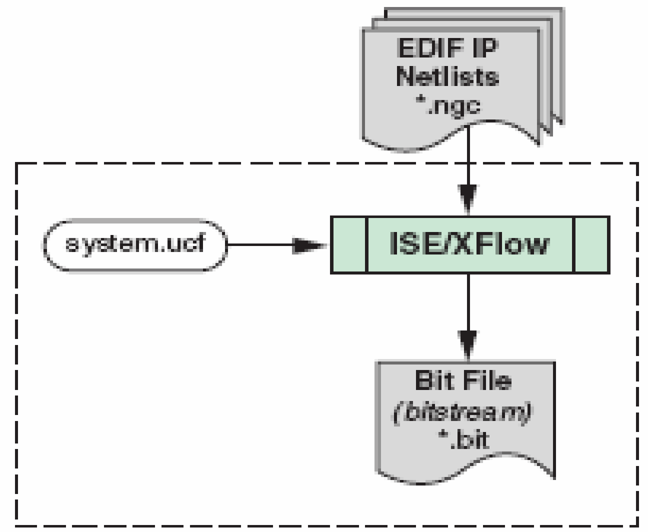
- **Platform Generator – PlatGen**
  - **Reads MHS & MPD Files**
  - Creates the synthesis, HDL, and implementation directories
  - Generates the HDL wrapper files for the peripherals & the top-level system
  - Calls XST to synthesize the HDL wrapper files
  - **Generates system netlist, peripheral netlists, and BMM file**



Hardware → Generate Netlist

Hardware Implementation Flow

- **XFlow – Implement Hardware & Generate Bitstream** (Automatic Approach)
  - **Reads .ngc netlists, system.bmm and system.ucf files**
  - Calls ISE Implementation tools using fast_runtime.opt
  - NGDBild, MAP, PAR & TRACE are executed
  - Then calls BitGen program using bitgen.ut file
  - **Generates system.bit & system_bd.bmm files**

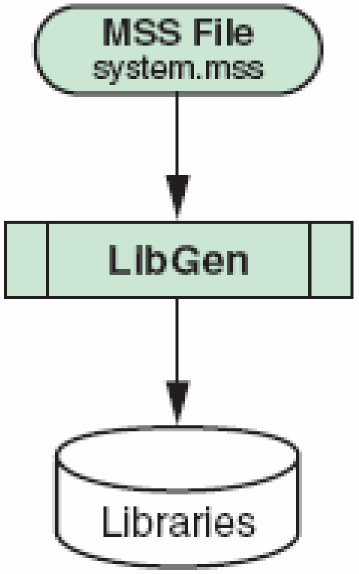


Hardware → Generate Bitstream

- Manual Hardware Implementation Flow using ISE GUI will be described later

Library Generation Flow

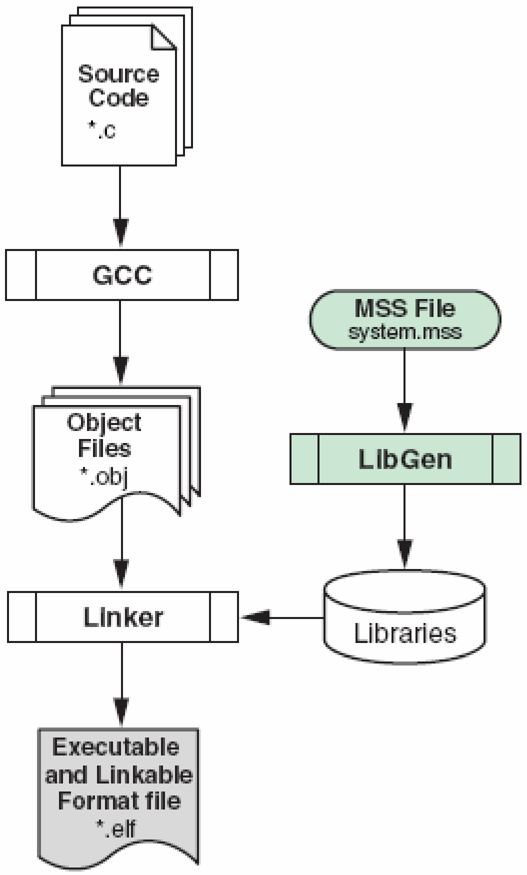
- **Library Generator – LibGen**
  - **Reads MSS file**
  - The MSS file defines the drivers associated with peripherals, standard input/output devices, interrupt handler routines, and other related software features
  - **Generates libraries and device drivers**
  - **Produces** an archive of object files:
    - **libc.a** - Standard C library
    - **libXil.a** - Xilinx library
    - **libm.a** - Math functions library



Software → Generate Libraries & BSPs

Software Application Flow

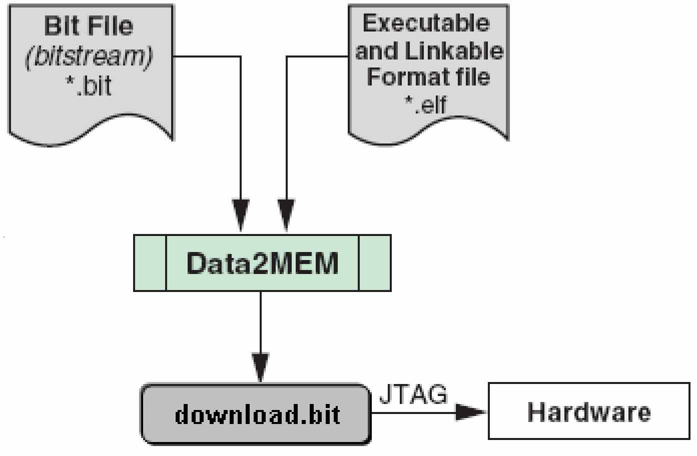
- **Compile program sources**
  - **Reads *.c, *.c++, *.h, libc.a, libXil.a, libm.a**
  - Compiles each software application and builds the executable files for each processor
  - Four stages:
    - *Pre-processor*: Replaces all macros with definitions as defined in the .c or .h files
    - *Compiler*: Machine-specific and language-specific - compiles C/C++ code
    - *Assembler*: Converts code to machine language and generates the object file
    - *Linker*: Links all the object files using user-defined or default linker script
  - **Generates ELF file**



Software → Build All User Applications

Merging Hardware and Software Flows

- **Data2MEM – Update the Bitstream**
  - **Reads *system_bd.bmm*, *system.bit*, *executable.elf***
  - Invokes the BitInit tool, which initializes the instruction memory of the processor
  - The instruction memory may be initialized with a bootloop, bootloader, or an actual application
  - **Generates download.bit file**



Device Configuration → Update Bitstream

Configuring the FPGA

- Download the Bitstream
  - Input file → *download.bit*
  - This downloads the *download.bit* file onto the target board using the Xilinx iMPACT tool in batch mode
  - XPS uses the *etc/download.cmd* file for downloading the Bitstream
  - The download.cmd file contains information such as the type of cable is used and the position of the FPGA in a JTAG chain

Device Configuration →Download Bitstream

Outline

- Introduction
- **EDK**
  - Overview of EDK
  - Embedded Development Design Flow
  - **Embedded Project Management**
- Supported Platforms

Modifying the Hardware

- Add cores, edit core parameters, and make bus and port connections through System Assembly view

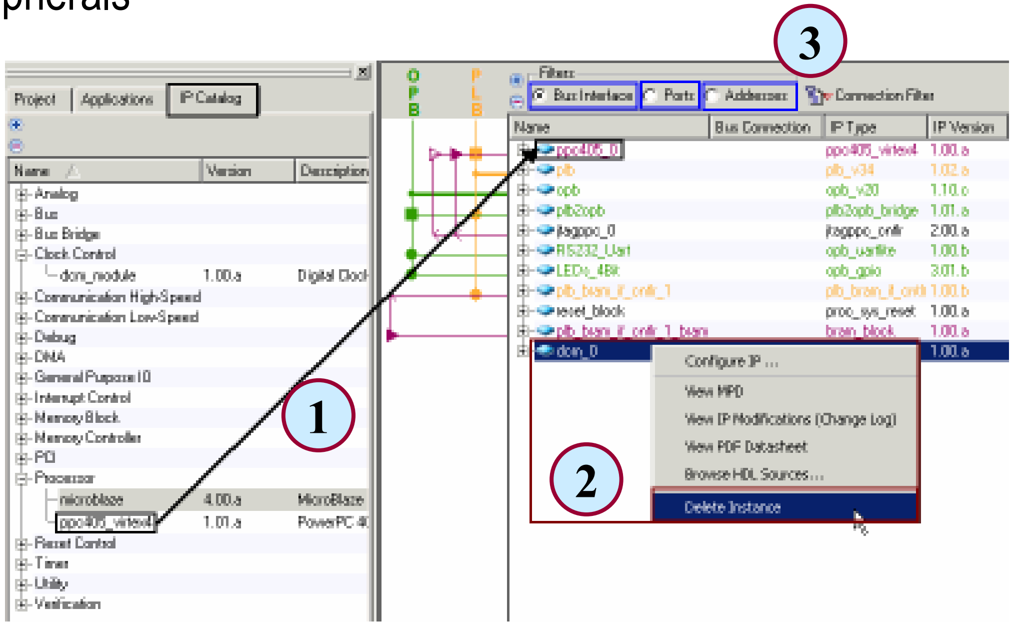
1 Select IP Catalog tab to add peripherals

- Select a core and drop it in the system view or double-click on it to add

2 In the System View select an instance, right click, and then select Delete Instance

3 Change settings using appropriate filters and select an instance

- Base and end addresses
- Parameters
- Ports



Name	Version	Description
Analog		
Bus		
Bus Bridge		
Clock Control		
dcm_module	1.00.a	Digital Clock
Communication High-Speed		
Communication Low-Speed		
Debug		
DMA		
General Purpose IO		
Interrupt Control		
Memory Block		
Memory Controller		
PCI		
Processor		
microblaze	4.00.a	MicroBlaze
ppc405_virtex4	1.01.a	PowerPC 40
Reset Control		
Timer		
Utility		
Verification		

Filters: ● Bus Interface ○ Ports ○ Addresses

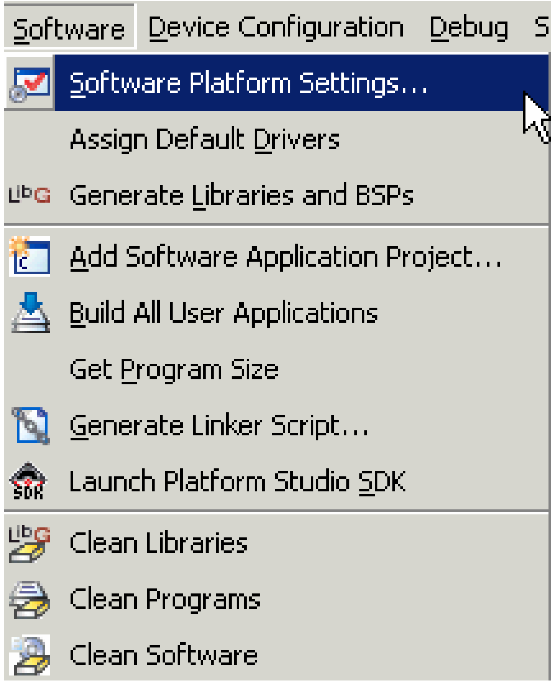
Name	Bus Connection	IP Type	IP Version
ppc405_0		ppc405_virtex4	1.00.a
plb		plb_v34	1.02.a
opb		opb_v20	1.10.c
plb2opb		plb2opb_bridge	1.01.a
jtagppc_0		jtagppc_cntlr	2.00.a
RS232_Uart		opb_uartlite	1.00.b
LEDs_4Bit		opb_gpio	3.01.b
plb_bram_if_cntlr_1		plb_bram_if_cntlr	1.00.b
reset_block		proc_sys_reset	1.00.a
plb_bram_if_cntlr_1_bram		bram_block	1.00.a
dcm_0			1.00.a

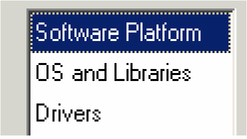
Software Application Management

- XPS supports test application creation and linker script management through BSB
- XPS lets you specify multiple application projects in the **Applications** tab
- XPS has an integrated editor for viewing and editing the C source and header files of the user program
- The source code is grouped for each processor instance. You can add or delete the list of source code files for each processor
- All of the source code files for a processor are compiled by using the compiler specified for that processor
- XPS tracks changes to C/C++ source files and recompiles when necessary
- Can launch the Platform Studio Software Development Kit (SDK)

Editing Software Settings

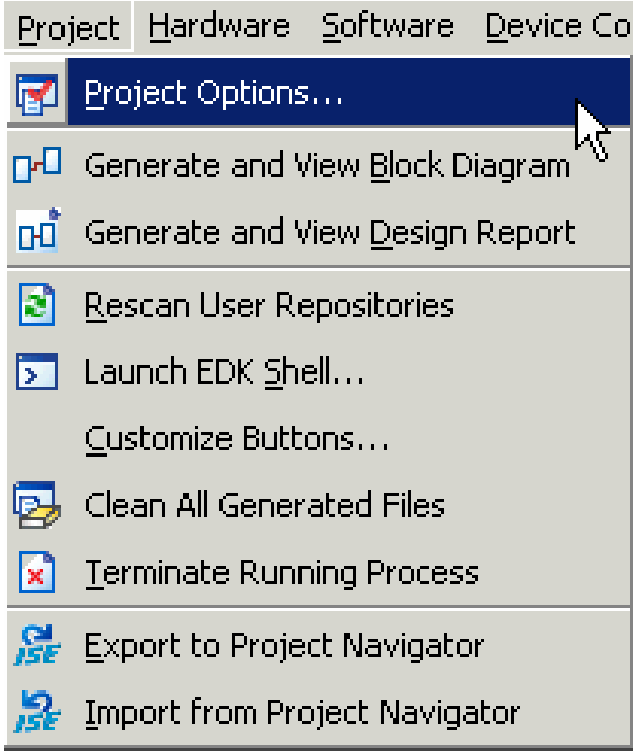
- Sets all of the software platform-related options in the design
- Has multiple forms selection:
  - Software Platform
    - CPU Driver
    - OS and OS Version selection
    - Libraries selection
    - Set core clock frequencies
  - OS and Libraries
    - Identify stdin and stdout devices
    - Configure OS and selected libraries
  - Drivers
    - Select drivers and versions
    - Core clock frequency


Software
Device Configuration
Debug
S
Software Platform Settings...
Assign Default Drivers
Generate Libraries and BSPs
Add Software Application Project...
Build All User Applications
Get Program Size
Generate Linker Script...
Launch Platform Studio SDK
Clean Libraries
Clean Programs
Clean Software



Setting Project Options

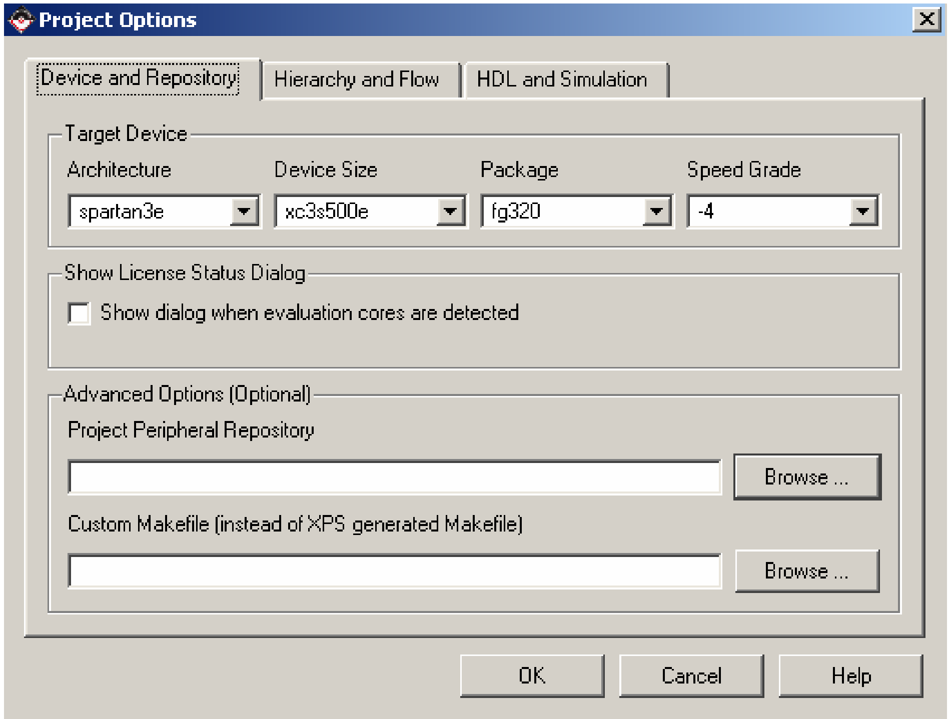
- XPS supports project options settings for:
  - Device and Repository tab
  - Hierarchy and Flow tab
  - HDL and Simulation tab


Project
Hardware
Software
Device Co
Project Options...
Generate and View Block Diagram
Generate and View Design Report
Rescan User Repositories
Launch EDK Shell...
Customize Buttons...
Clean All Generated Files
Terminate Running Process
Export to Project Navigator
Import from Project Navigator

Project Options

Device and Repository Tab

- Set/Change Target Device
  - Architecture
  - Device Size
  - Package
  - Speed Grade
- Peripheral Repository Directory
  - Provide path to custom IP not present in the current project directory structure
- Custom Makefile Directory



Project Options

Device and Repository | Hierarchy and Flow | HDL and Simulation

Target Device

Architecture	Device Size	Package	Speed Grade
spartan3e	xc3s500e	fg320	-4

Show License Status Dialog

□ Show dialog when evaluation cores are detected

Advanced Options (Optional)

Project Peripheral Repository

Custom Makefile (instead of XPS generated Makefile)

Note: Detailed information on the Hierarchy and Flow tab is provided in the “Adding Your Own IP”

Outline

- Introduction
- EDK
  - Overview of EDK
  - Embedded Development Design Flow
  - XPS Platform Management
- **Supported Platforms**

Supported Platforms

- Operating systems
  - Windows XP 32-bit SP2 Professional
  - Windows Vista 32-bit
  - Linux Red Hat Enterprise (5.0 32-bit/64-bit, 4.0 32-bit)
- FPGA families
  - Spartan™-II/IIE (MicroBlaze™ processor)
  - Spartan-3/3E (MicroBlaze processor)
  - Spartan-3A/AN (MicroBlaze processor)
  - Spartan-3A DSP
  - Virtex™ and Virtex-E (MicroBlaze processor)
  - Virtex-II (MicroBlaze processor)
  - Virtex-II Pro (MicroBlaze and PowerPC™ processors)
  - Virtex-4 FX (MicroBlaze and PowerPC processors) and LX/SX (MicroBlaze processor)
  - Virtex-5 LX, LXT, SXT (MicroBlaze processor)

BSB-Supported Platforms

- A list of supported Xilinx hardware boards:
  - Xilinx AFX board
  - Spartan-3/3E/3A/3AN Starter Kits
  - Spartan-3E 1600E MicroBlaze Development Kit
  - Xilinx/Lyrtech Spartan-3A DSP 3400
  - Virtex-5 ML501, ML505, ML506
  - Virtex-4 ML401, ML402, ML403, ML405, ML410
  - Custom board
- Board definition (.xbd) files for third party boards can be downloaded from the board vendor web site
  - Links from the BSB wizard and Xilinx embedded Web page

Knowledge Check

- What is the MHS file?
- What does the PlatGen tool do?
- What tool is used to place executable code in an FPGA block RAM?

Answers

- What is the MHS file?
  - The MHS file is the Microprocessor Hardware Specification; it specifies processors, hardware peripherals, bus connections, and address spaces for the hardware
- What does the PlatGen tool do?
  - PlatGen takes the MHS file and creates the system and peripheral netlists, HDL wrapper files, BMM file, etc.
- What tool is used to place executable code in an FPGA block RAM?
  - The BitInit tool will take the BMM file and create the proper initialization for the block RAM that is assigned to the executable memory space

Knowledge Check

- How can you add or change configuration settings once the hardware system is build?

- What does the LibGen tool do?

- What is the difference between system.bit and download.bit files?

Answers

- How can you add or change configuration settings once the hardware system is build?
  - Select IP Catalog tab, expand related IP peripheral folder, select a desired IP, and double-click on it to add it to the design
  - Select an IP instance in the System Assembly View panel, right click on it, and select desired configuration
- What does the LibGen tool do?
  - Read MSS file and generate libraries
- What is the difference between system.bit and download.bit files?
  - The system.bit file contains only hardware description whereas download.bit file contains both hardware description as well as executable software

Where Can I Learn More?

- Tool documentation
  - *Getting Started with the Embedded Development Kit*
  - *Processor IP Reference Guide*
  - *Embedded Systems Tools Guide*
  - *Xilinx Drivers*

- Processor documentation
  - *PowerPC Processor Reference Guide*
  - *PowerPC 405 Processor Block Reference Guide*
  - *MicroBlaze Processor Reference Guide*

- Support Website
  - EDK Website: www.xilinx.com/edk

XILINX®