ASIC Logic

Speaker: Juin-Nan Liu

Adopted from National Chiao-Tung University
IP Core Design

Goal of This Lab

- ❑ Prototyping
- ❑ Familiarize with ARM Logic Module (LM)
- ❑ Know how to program LM

Outline

- ❑ ***Introduction***
- ❑ ARM System Overview
- ❑ Prototyping with Logic Module
- ❑ Lab – ASIC Logic

Introduction

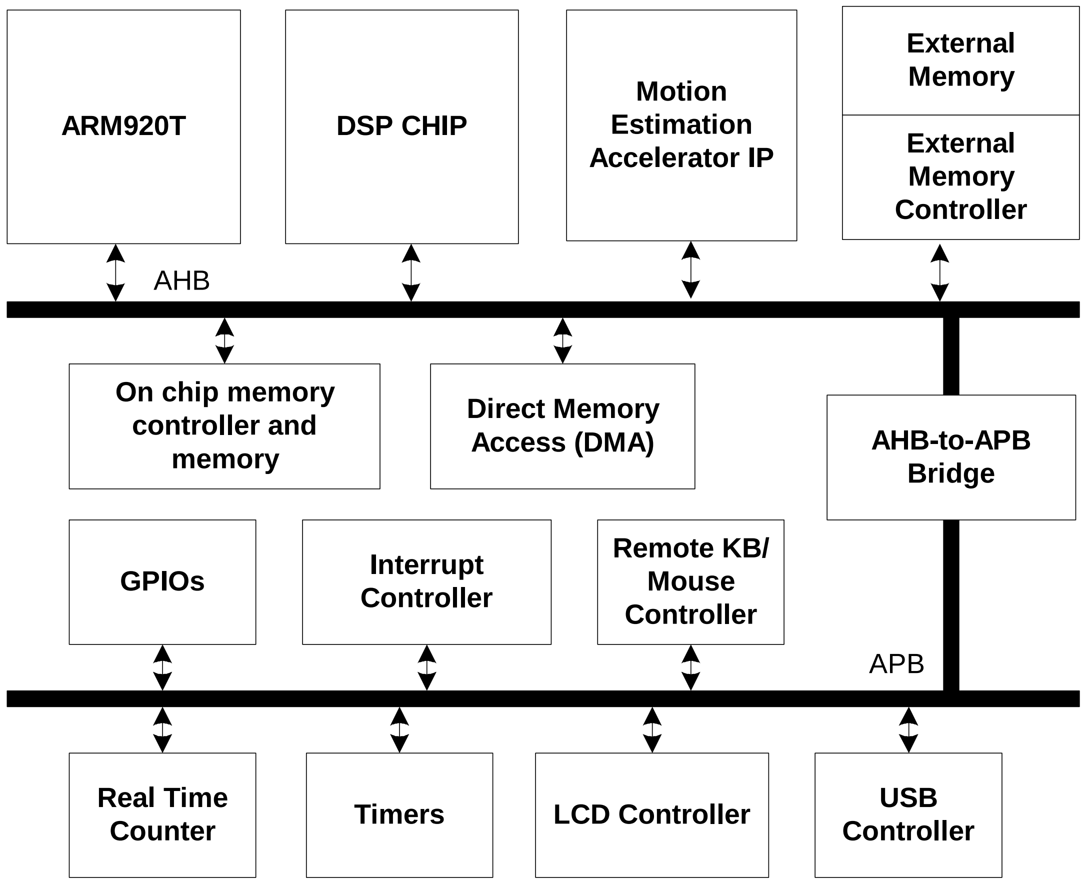
- ❑ Rapid Prototyping – A fast way to verify your prototype design.
  - – Enables you to discover problems before tape out.
  - – Helps to provide a better understanding of the design’s behavior.
- ❑ ARM Integrator and Logic Module can be used for Hardware Design Verification and HW/SW co-verification.
  - – Hardware Design Verification: using LM stand alone.
  - – HW/SW co-verification: using LM, CM, Integrator together.

Outline

- ❑ Introduction
- ❑ ***ARM System Overview***
  - – ARM Synchronization Scheme: Interrupt
  - – ARM Synchronization Scheme: Polling
- ❑ Prototyping with Logic Module
- ❑ Lab – ASIC Logic

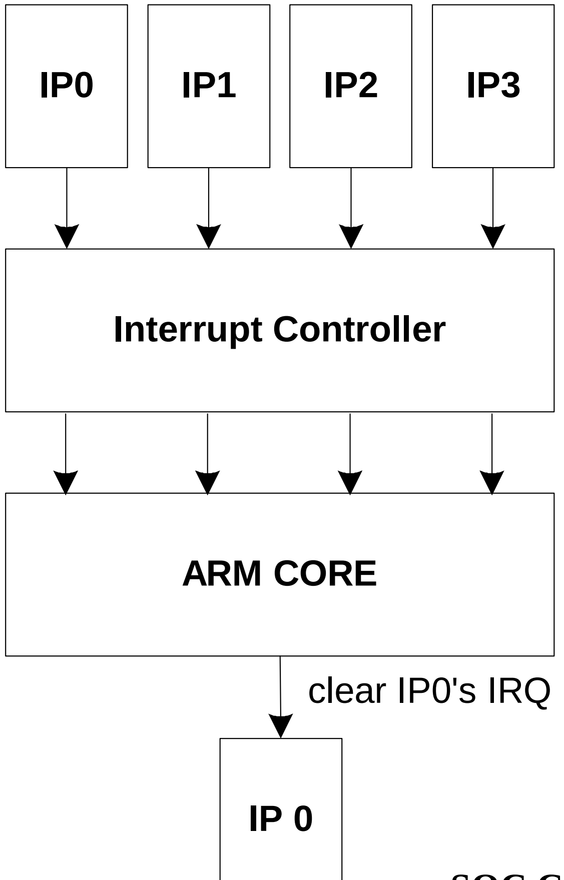
ARM System Overview

- A typical ARM system consists of an ARM core, a DSP chip for application-specific needs, some dedicated hardware accelerator IPs, storages, and some peripherals and controls.


ARM920T
DSP CHIP
Motion Estimation Accelerator IP
External Memory
External Memory Controller
AHB
On chip memory controller and memory
Direct Memory Access (DMA)
AHB-to-APB Bridge
GPIOs
Interrupt Controller
Remote KB/ Mouse Controller
APB
Real Time Counter
Timers
LCD Controller
USB Controller

ARM System Synchronization Scheme: Interrupt

- A device asserts an interrupt signal to request the ARM core handle it.
- The ARM core can perform tasks while the device is in use.
- Needs Interrupt Controller. More hardware.



IP0, IP1, IP2, and IP3 raised interrupt request (IRQ) at the same time. The IRQs are sent to the interrupt controller.

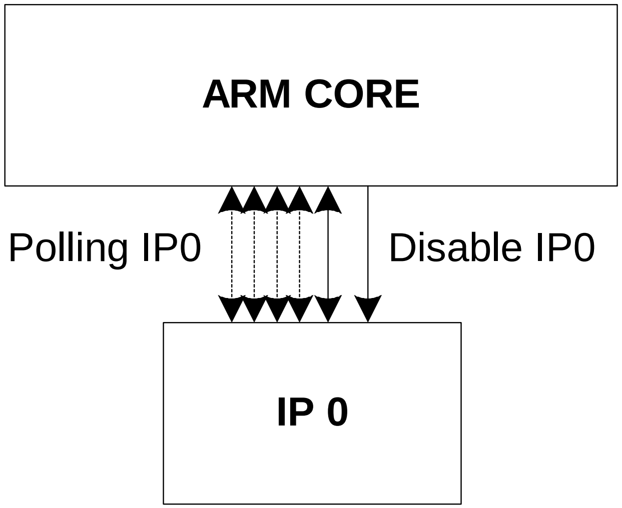
Interrupt controller receives the IRQs and update the IRQ status indicating the IRQ sources.

ARM core receives the IRQs, deteremines which IRQ should be handled according to programmed priorities. and then executes the corresponding interrupt service routine (ISR).

The ISR performs its operations and clears the IP0's interrupt.

ARM System Synchronization Scheme: Polling

- The ARM core keeps checking a register indicating if the device has done its task.
- The ARM core is busy “polling” the device while the device is in use.
- Less hardware.



ARM core polls IP0's ready register after IP0 has been enabled.

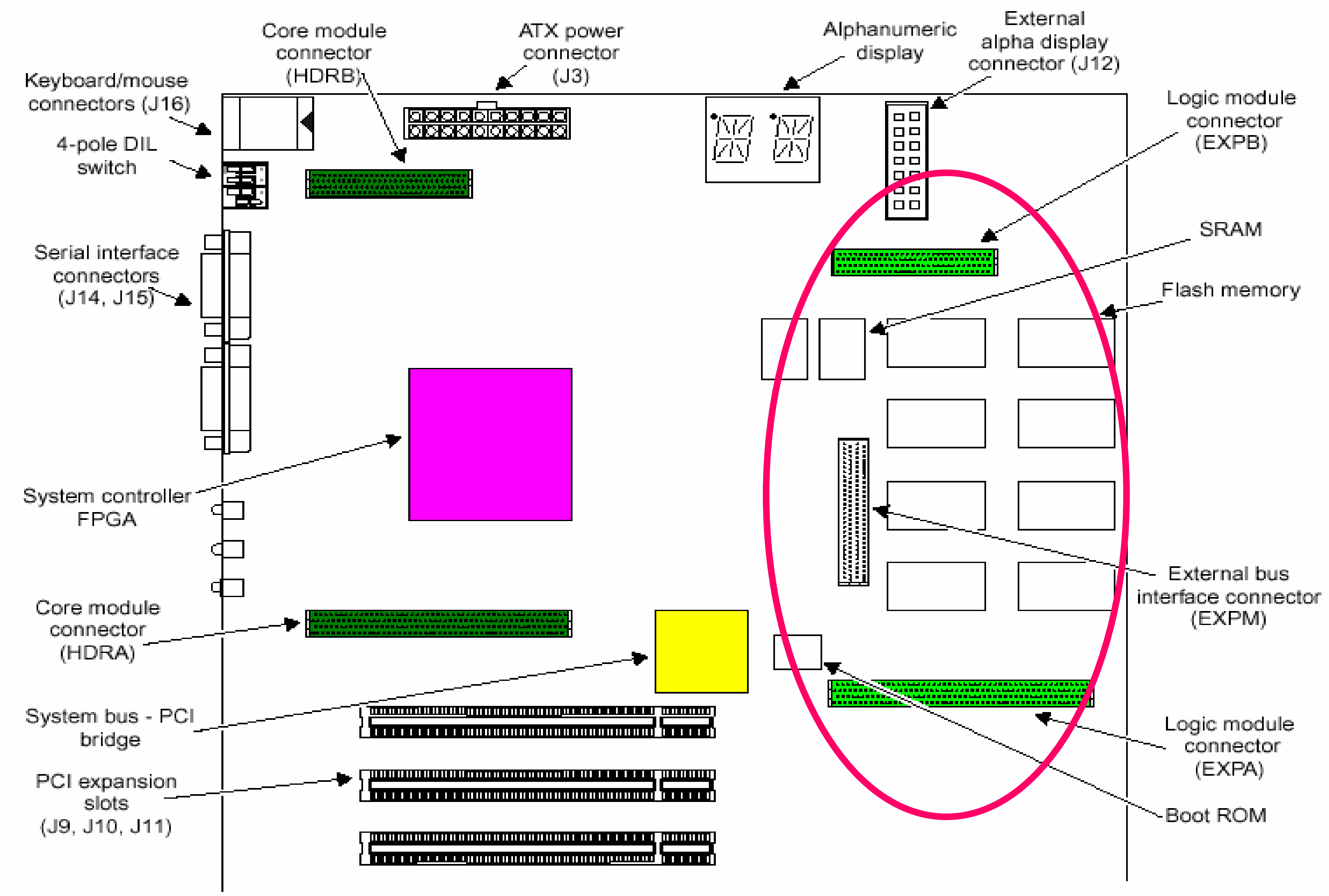
Once IP0 is done with its operation, ARM core will know from the changed value of the ready register.

ARM core will execute the corresponding operations and then disable IP0.

Outline

- ❑ Introduction
- ❑ ARM System Overview
- ❑ ***Prototyping with Logic Module***
  - – ARM Integrator AP & ARM LM
  - – FPGA tools
  - – Example 1
  - – Example 2
  - – Exercise
- ❑ Lab – ASIC Logic

AP Layout



Core module connector (HDRB)
ATX power connector (J3)
Alphanumeric display
External alpha display connector (J12)
Keyboard/mouse connectors (J16)
4-pole DIL switch
Logic module connector (EXPB)
SRAM
Flash memory
Serial interface connectors (J14, J15)
System controller FPGA
External bus interface connector (EXPM)
Core module connector (HDRA)
System bus - PCI bridge
Logic module connector (EXPA)
PCI expansion slots (J9, J10, J11)
Boot ROM

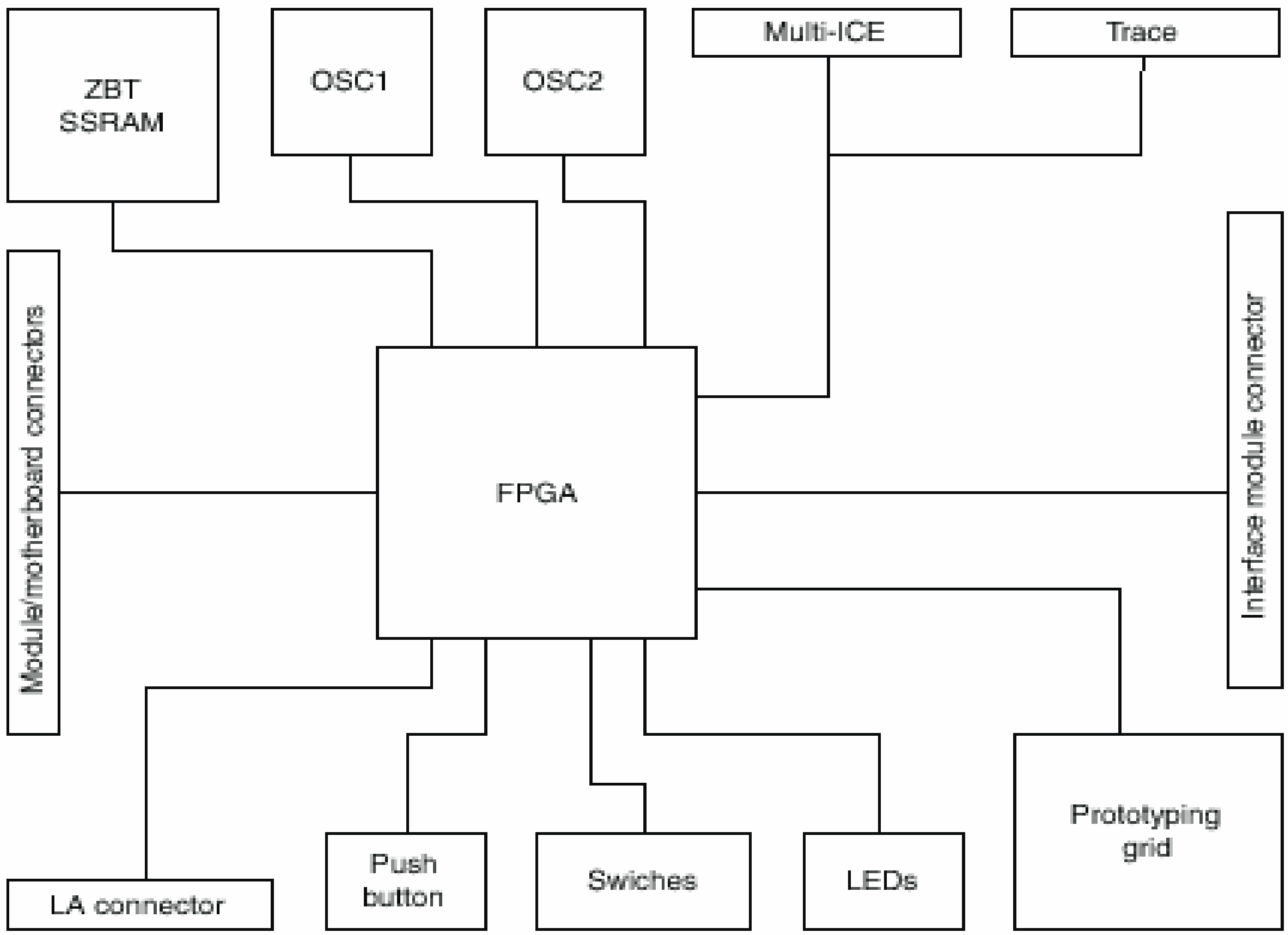
What is LM

- *Logic Module*
- A platform for developing ***Advanced Microcontroller Bus Architecture*** (AMBA), ***Advanced System Bus*** (ASB), ***Advanced High-performance Bus*** (AHB), and ***Advanced Peripheral Bus*** (APB) peripherals for use with ARM cores.

Using the LM

- It can be used in the following ways:
  - As a standalone system
  - With an CM, and a AP or SP motherboard
  - As a CM with either AP or SP motherboard if a synthesized ARM core is programmed into the FPGA
  - Stacked without a motherboard, if one module in the stack provides system controller functions of a motherboard

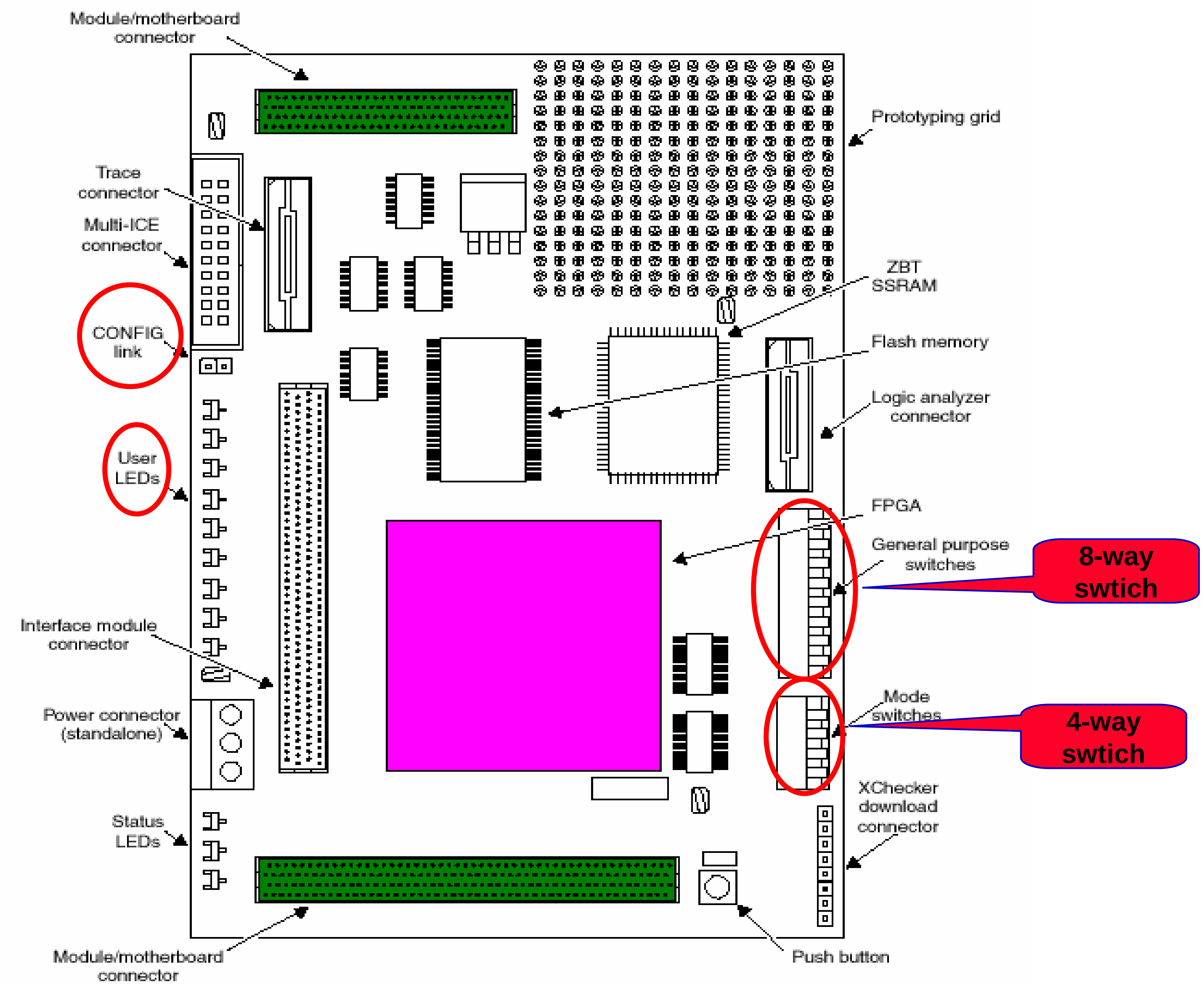
LM Architecture



Components of LM

- Altera or ***Xilinx*** FPGA
- Configuration PLD and flash memory for storing FPGA configurations
- 1MB ZBT SSRAM
- Clock generators and reset sources
- A 4-way flash image selection switch and an 8-way user definable switch
- 9 user-definable surface-mounted LEDs (8G1R)
- User-definable push button
- Prototyping grid
- System bus connectors to a motherboard or other modules

LM Layout

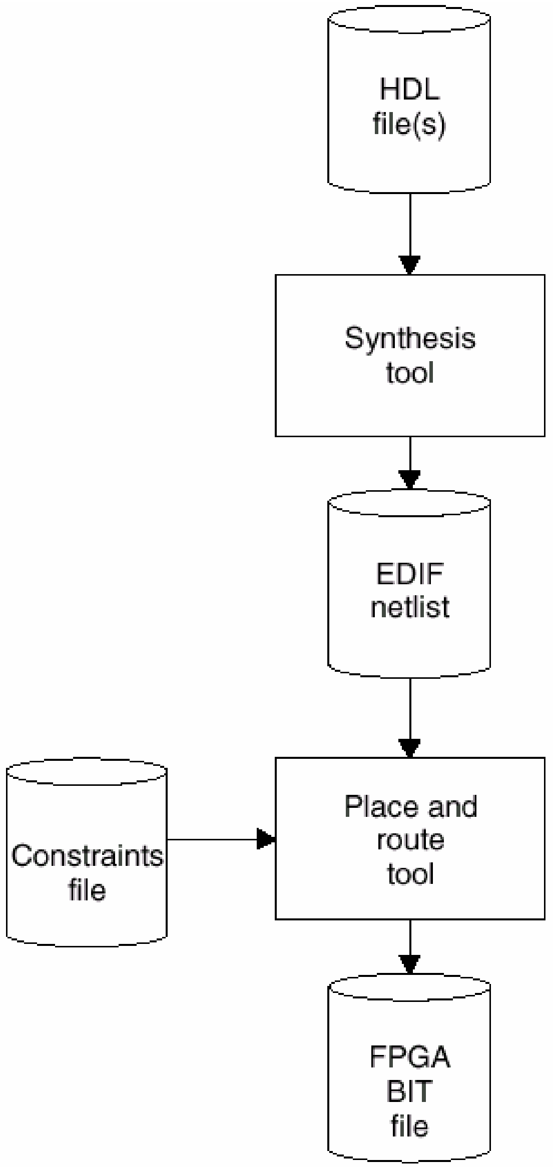

Module/motherboard connector
Prototyping grid
Trace connector
Multi-ICE connector
CONFIG link
ZBT SSRAM
Flash memory
User LEDs
Logic analyzer connector
FPGA
General purpose switches
8-way swtich
Interface module connector
Mode switches
4-way swtich
Power connector (standalone)
XChecker download connector
Status LEDs
Module/motherboard connector
Push button

Links

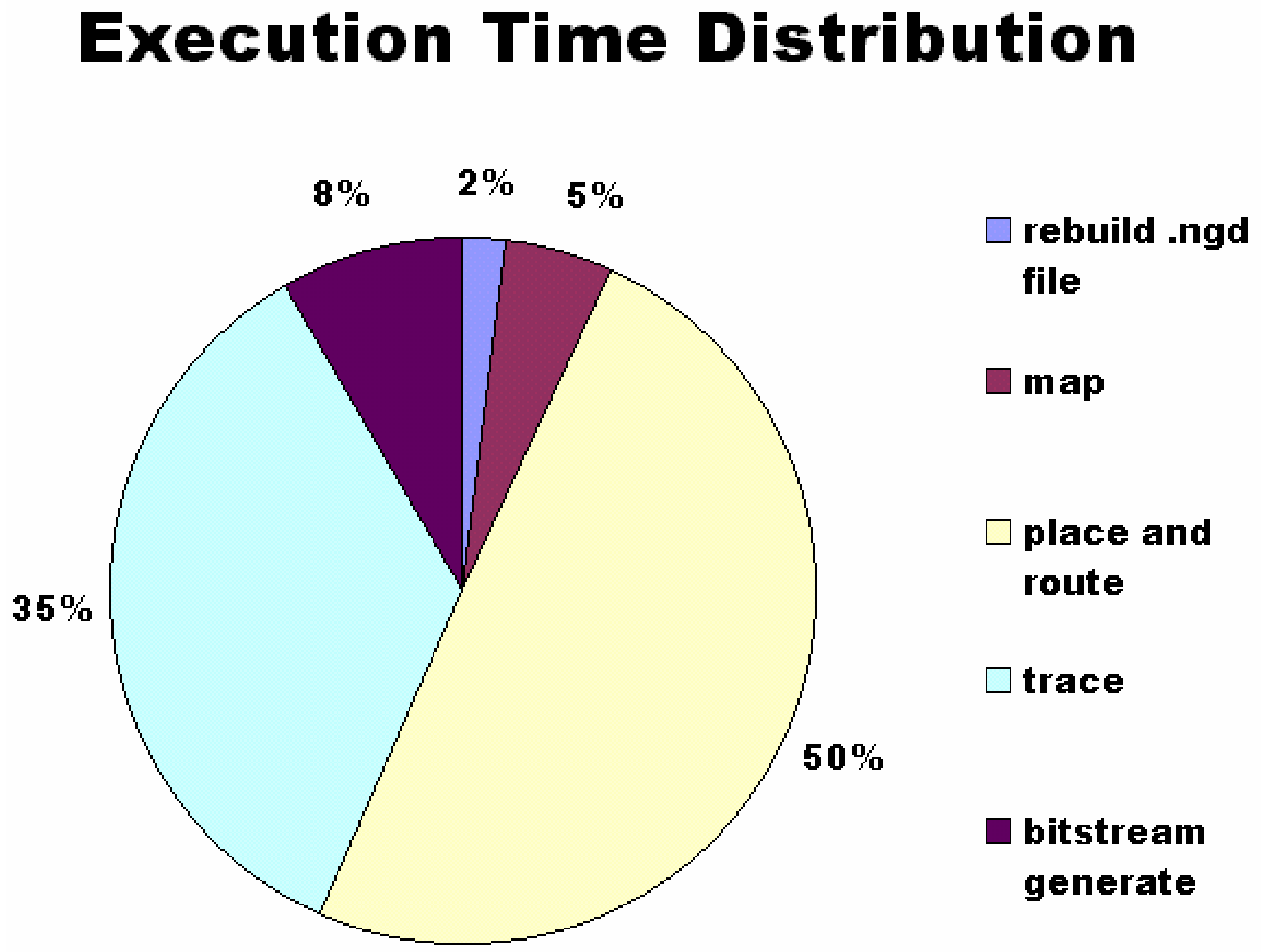
- CONFIG link
  - Enable ***configuration mode***, which changes the JTAG signal routing and is used to ***download new PLD or FPGA configurations***.
- JTAG, Trace, and logic analyzer connectors
- Other links, switches, and small ICs can be added to the prototyping grid if required.

FPGA tools

Xilinx GUI Synthesis Tool



A Timing Information Example

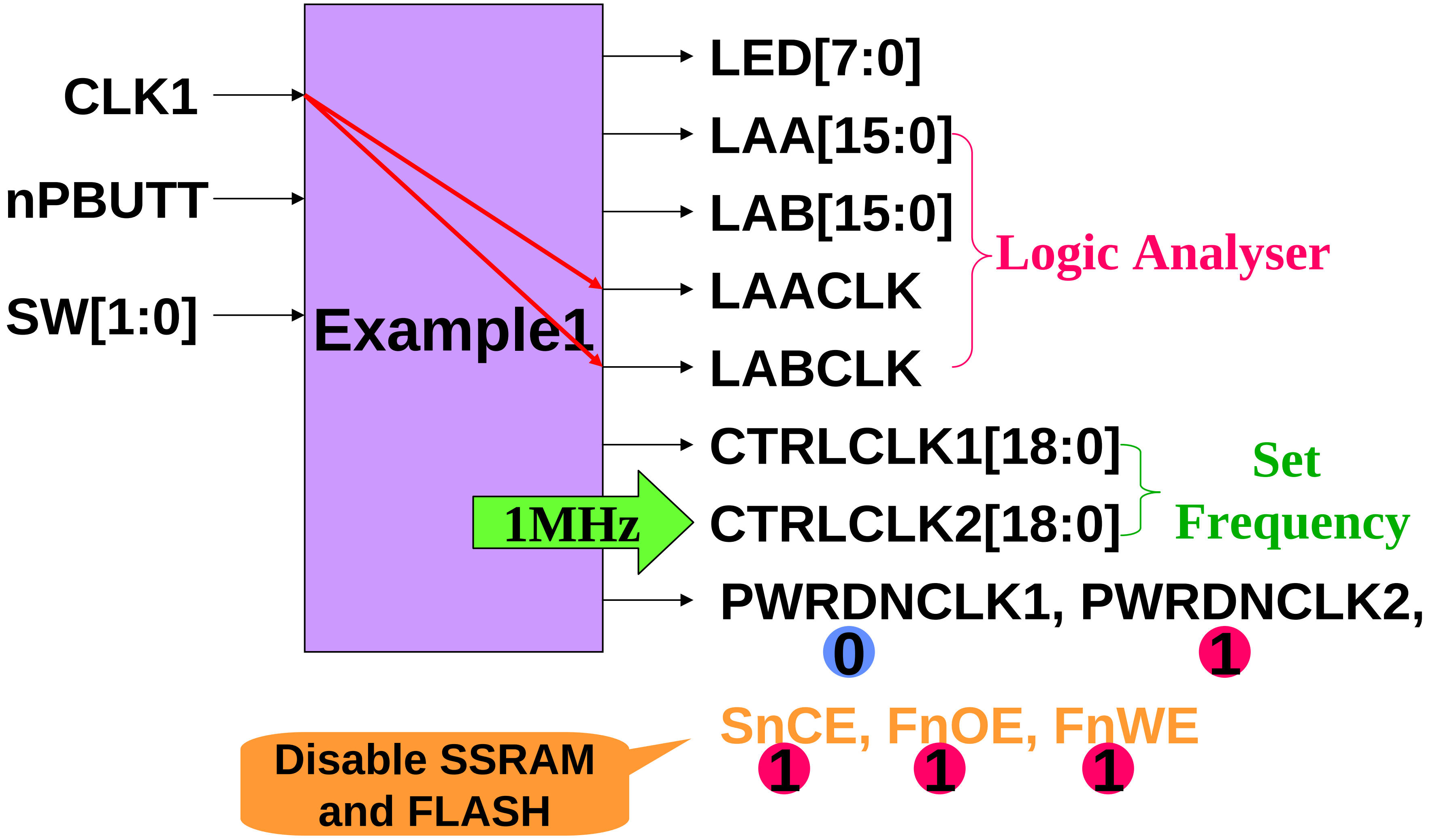


<span style="color:#e6007e">We **strongly suggest** you to perform place-and-route operation on a better PC, it’s a very time-consuming work!</span>

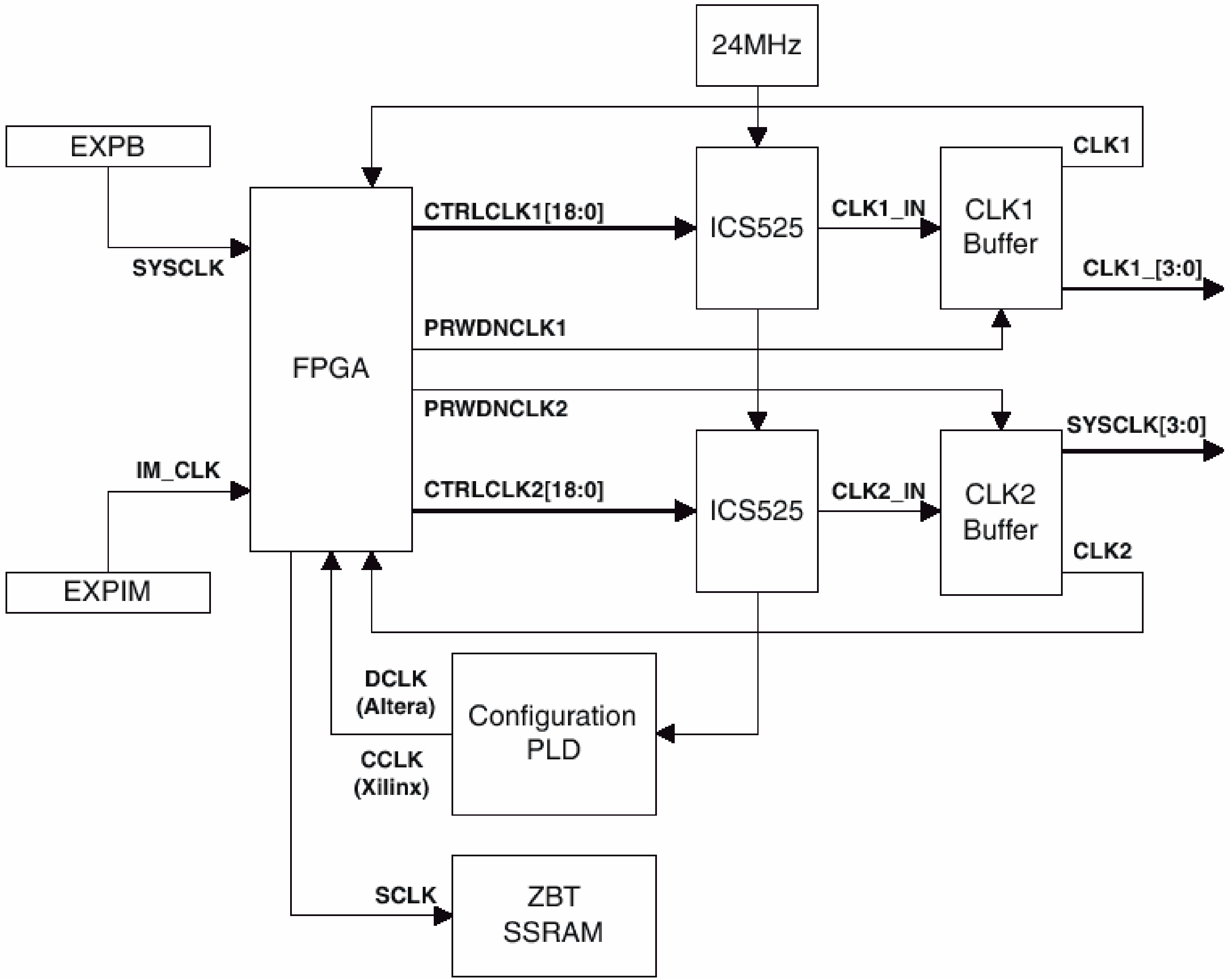
Example 1

- Path = .\lab5\Codes\HW\example1\
- Count up on logic analyzer channel a
- Count down on logic analyzer channel b
- Reset by pushbutton
- LEDs scan from left to right.
- Switches [0:1] (brown:red) control the clock frequency (***CTRLCLK1***) and affect the LEDs scanning frequency.

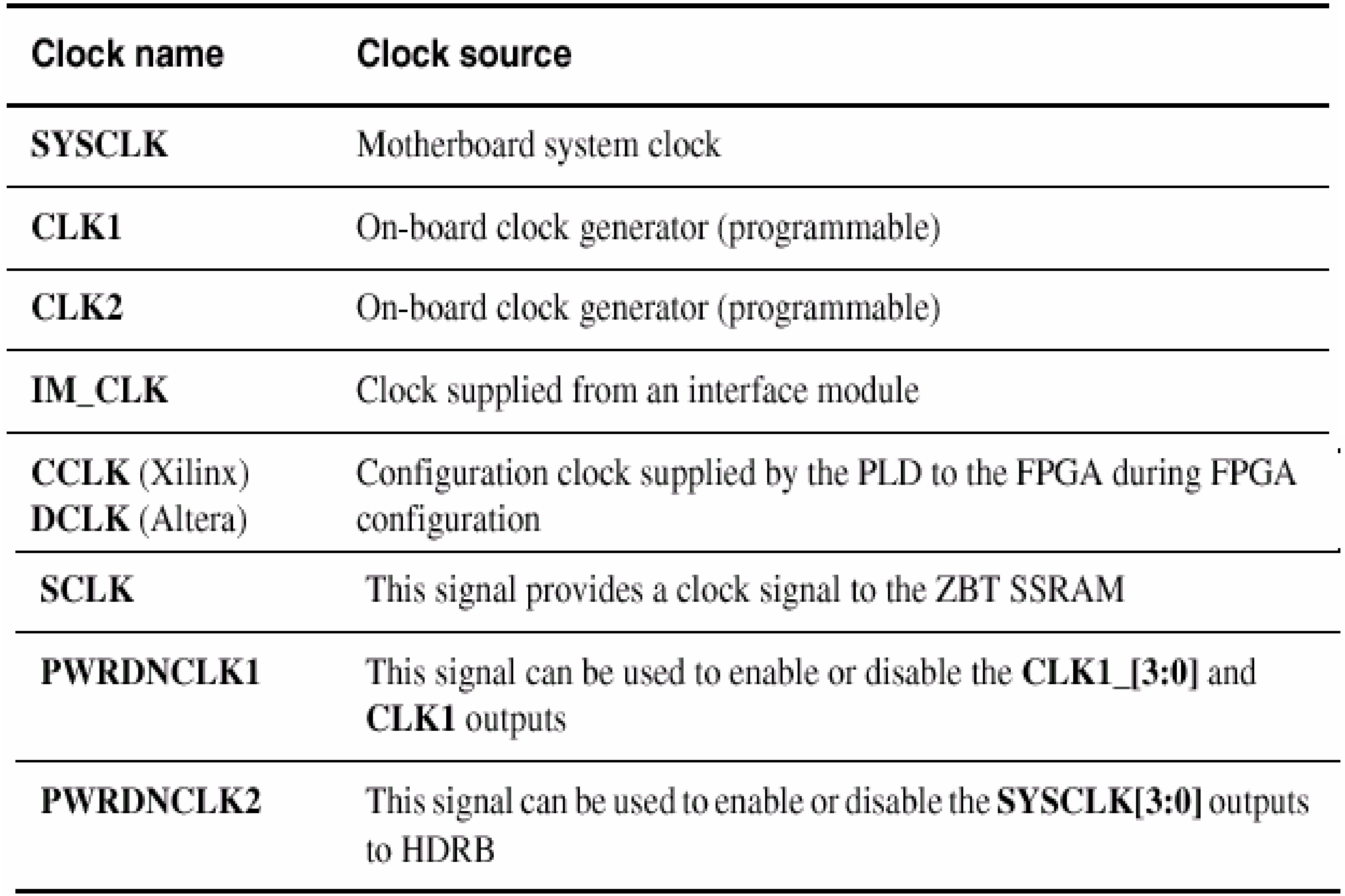
Example 1 (cont.)



On-board Clock Generators



Clock Signal Summary



Clock name	Clock source
SYSCLK	Motherboard system clock
CLK1	On-board clock generator (programmable)
CLK2	On-board clock generator (programmable)
IM_CLK	Clock supplied from an interface module
CCLK (Xilinx) **DCLK** (Altera)	Configuration clock supplied by the PLD to the FPGA during FPGA configuration
SCLK	This signal provides a clock signal to the ZBT SSRAM
PWRDNCLK1	This signal can be used to enable or disable the **CLK1_[3:0]** and **CLK1** outputs
PWRDNCLK2	This signal can be used to enable or disable the **SYSCLK[3:0]** outputs to HDRB

Programming the LM Clock

Signals	Control parameter	Label
CTRLCLKx[18:16]	Output divider	S[2:0]
CTRLCLKx[15:9]	Reference divider	R[6:0]
CTRLCLKx[8:0]	VCO divider	V[8:0]

S	S[2:0]
2	001
4	011
5	100
6	111
7	101
8	010
9	110
10	000

$$\text{Frequency} = 48\text{MHz} \cdot \frac{(V[8:0] + 8)}{(R[6:0] + 2) \cdot S}$$

1MHz: CTRLCLKx=19'b1100111110000000100
2MHz: CTRLCLKx=19'b1100011110000000100
5MHz: CTRLCLKx=19'b1100001110000000111
10MHz: CTRLCLKx=19'b1100000110000000111

Constraint:

$$10\text{MHz} < 48\text{MHz} \cdot \frac{(V[8:0] + 8)}{(R[6:0] + 2)}$$

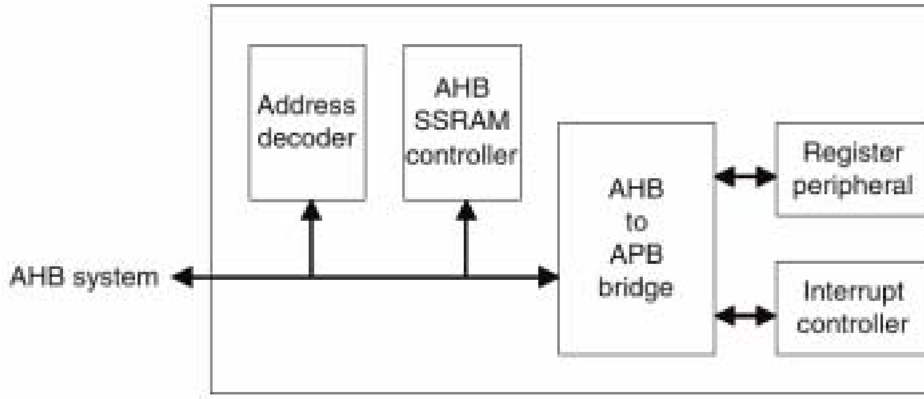
$$R[6:0] < 118$$

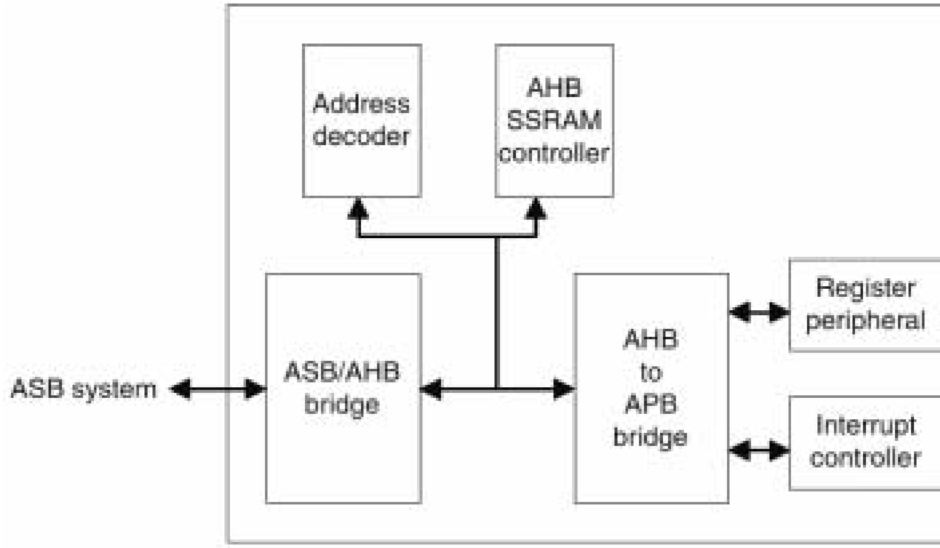
Example 2

- The example code operates as follows:
  1. Determines DRAM size on the core module and sets up the system controller
  2. Checks that the logic module is present in the AP expansion position
  3. Reports module information
  4. Sets the logic module clock frequencies
  5. Tests SSRAM for word, halfword, and byte accesses.
  6. Flashes the LEDs
  7. Remains in a loop that displays the switch value on the LEDs

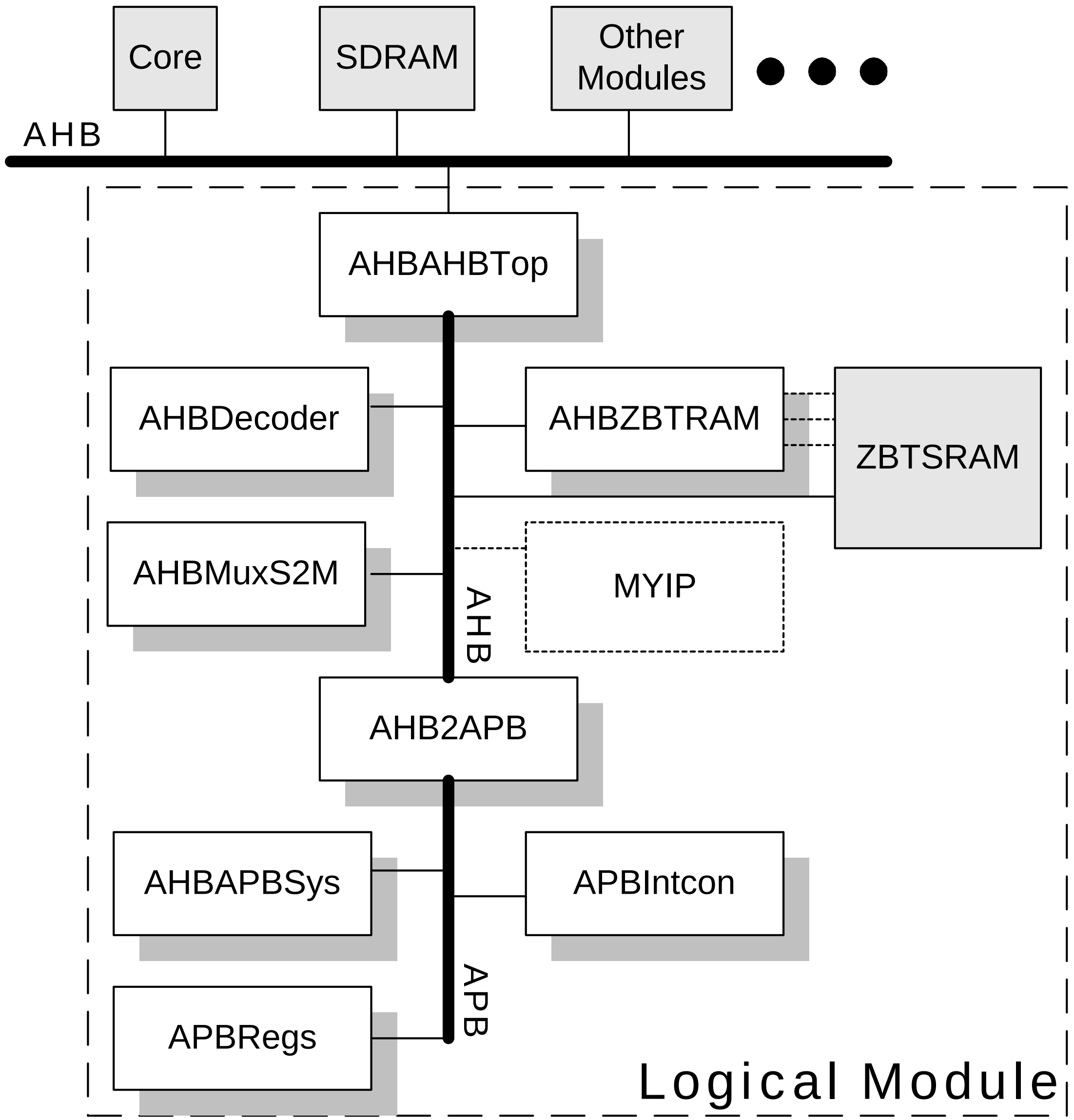
Two Platform – AHB & ASB

- ❑ Two versions of example 2 are provided to support the following implementations:
  - – AHB motherboard and AHB peripherals
  - – ASB motherboard and AHB peripherals
- ❑ Which AMBA has been downloaded on board can be observed by the alphanumber display
  - – **H**: AHB
  - – **S**: ASB





AHB Platform



Example2 Files Description

- Hardware files .*\Lab7\Codes\HW\example2\Verilog*
  - AHBAHBTop.v
  - AHBDecoder.v
  - AHBMuxS2M.v
  - AHBZBTRAM.v
  - AHB2APB.v
  - AHBAPBSys.v
  - APBIntCon.v
  - APBRegs.v

For Xilinx synthesis tool to generate lmxcv600e_72c_xcv2000e_via_reva_build0.bit

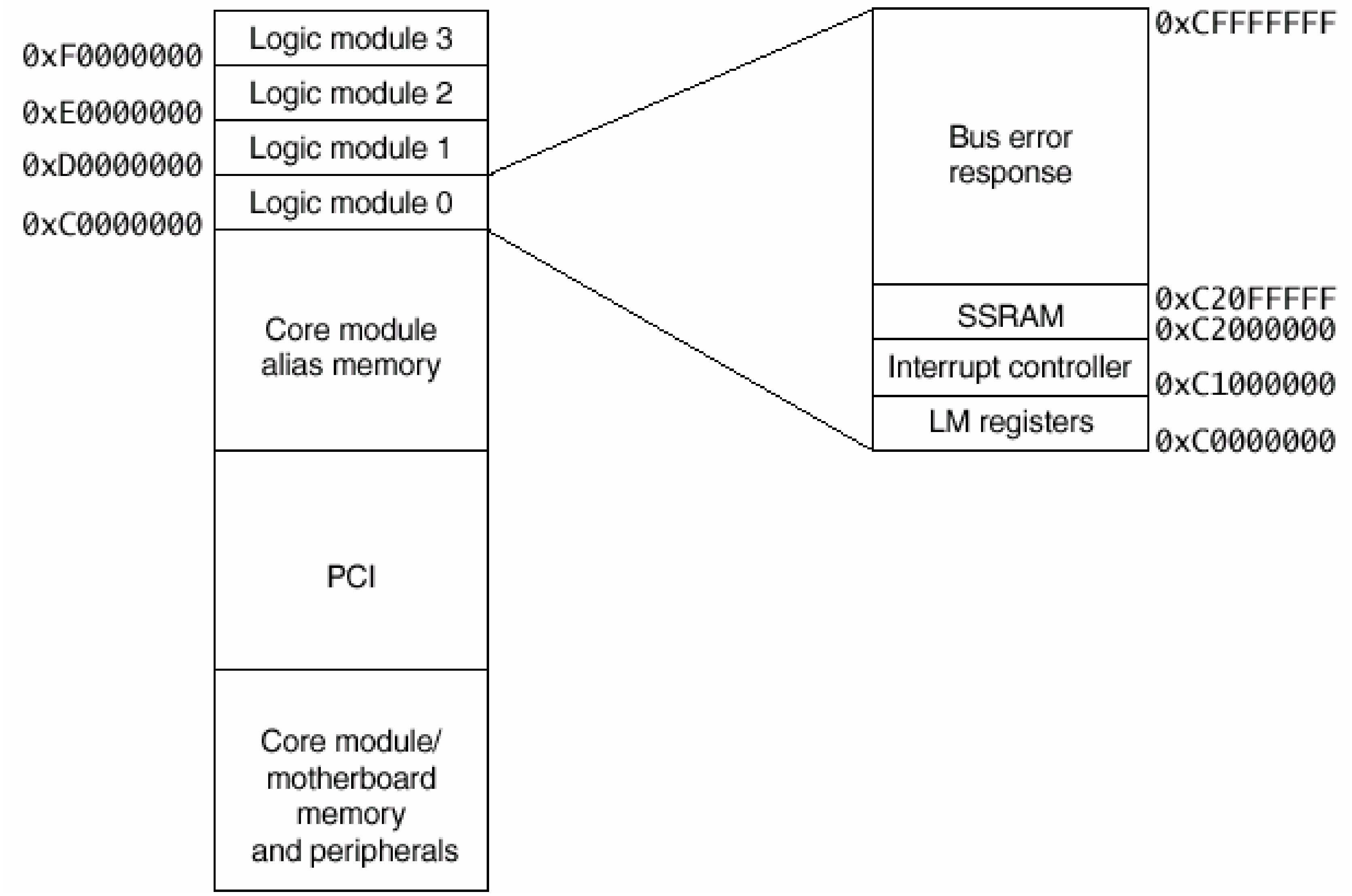
- Software program files .*\Lab7\Codes\SW\example2\*
  - sw.mcp
  - logic.c
  - logic.h
  - platform.h
  - rw_support.s

For codewarrior to generate sw.axf

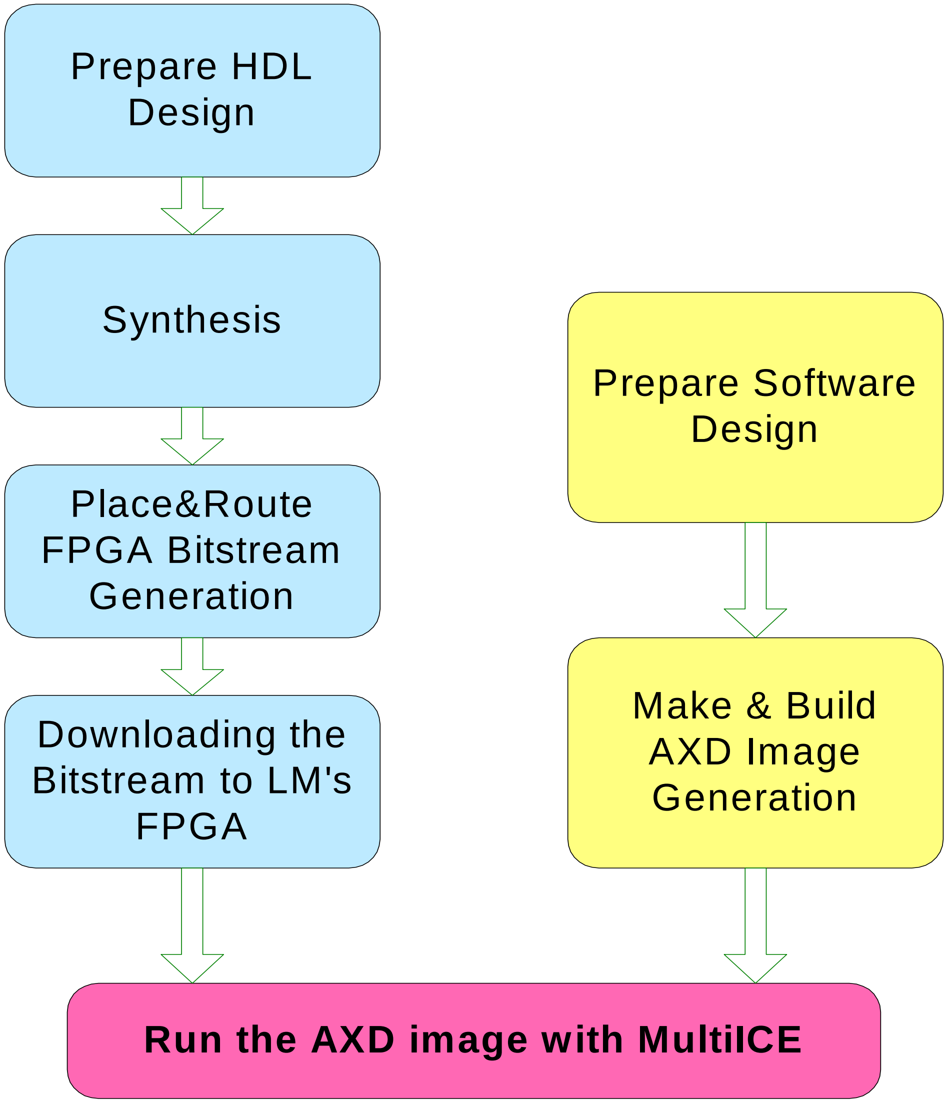
Software Description

- 5 files included in .*Lab7\Codes\SW\example2\*
  - sw.mcp: project file
  - logic.c: the main C code
  - logic.h: constant definitions
  - platform.h: constant definitions
  - rw_support.s: assembly functions for SSRAM testing

Integrator Memory Map



Work Flow Summary


Prepare HDL Design
Synthesis
Place&Route FPGA Bitstream Generation
Downloading the Bitstream to LM's FPGA
Prepare Software Design
Make & Build AXD Image Generation
Run the AXD image with MultiICE

Outline

- ❑ Introduction
- ❑ ARM System Overview
- ❑ Prototyping with Logic Module
- ❑ ***Lab – ASIC Logic***

Lab 5: ASIC Logic

- Goal
  - HW/SW Co-verification using Rapid Prototyping
- Principles
  - Basics and work flow for prototyping with ARM Integrator
  - Target platform: AMBA AHB sub-system
- Guidance
  - Overview of examples used in the Steps
- Steps
  - Understand the files for the example designs and FPGA tool
  - Steps for synthesis with Xilinx ISE 5.1i/5.2i
- Requirements and Exercises
  - RGB-to-YUV converting hardware module. See next slide
- Discussion
  - In example 1, explain the differences between the Flash version and the FPGA one.
  - In example 1, explain how to move data from DRAM to registers in MYIP and how program access these registers.
  - In example2, draw the interconnect among the functional units and explain the relationships of those interconnect and functional units in AHB sub-system
  - Compare the differences of polling and interrupt mechanism

Exercise: RGB to YCrCb Converter

- Convert the *rgb2ycrcb()* into hardware module and implement it on the ARM development system. Evaluate the improvement.
  - Hint: you may modify **ahbahbtop.v**, **ahbdecoder.v**, **ahbmuxs2m.v**, and **ahbzbtram.v** files in example2

Function:

$$\begin{bmatrix} Y \\ C_B \\ C_R \end{bmatrix} = \begin{bmatrix} 0.257 & 0.504 & 0.098 \\ -0.148 & -0.291 & 0.439 \\ 0.439 & -0.368 & -0.071 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix}$$

Reference

[1] http://twins.ee.nctu.edu.tw/courses/ip_core_02/index.html

[1] LM-XCV2000E.pdf

[2] DUI0098B_AP_UG.pdf

[3] progcards.pdf