

# HPS+FPGA Systems on DE1-SoC Board

Cristinel Ababei  
Marquette University

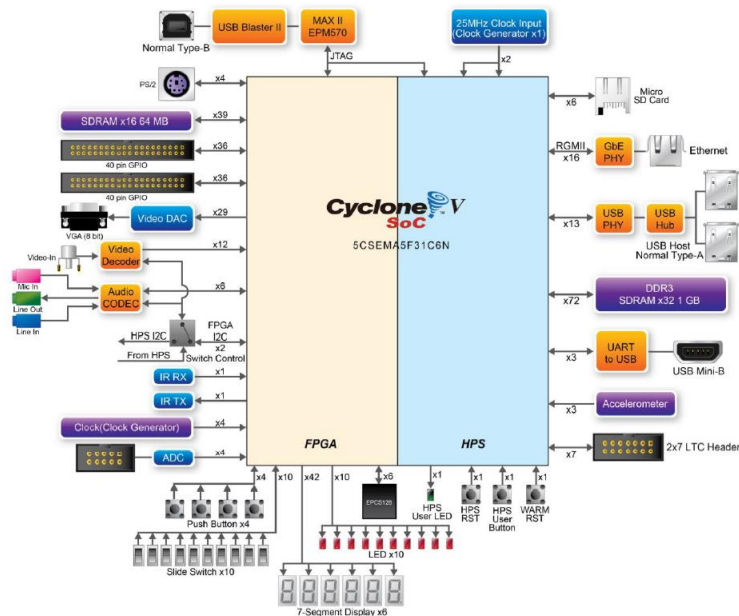
1

## HPS+FPGA Systems

- Projects that use both the Hardware Processor System (HPS) and the FPGA component
- Software needed (on Windows 10):
  - Quartus Prime Lite Edition
    - Use Platform Designer (used to be called Qsys) to instantiate and connect HPS component(s). **Note: Earlier the NIOS II softcore processors were used instead of HPS.**
    - Synthesize FPGA project and program FPGA
  - Intel SoC FPGA Embedded Development Suite (SoC EDS) – Standard Edition
    - Comprehensive tool suite for embedded software development on Intel FPGA SoC devices
    - Write C code, compile, copy executable on microSD card, and execute
  - Arm Development Studio (DS) - Intel SoC FPGA Edition
    - Arm Development Studio is an embedded C/C++ development toolchain designed specifically for Arm-based SoCs, from tiny microcontrollers to custom multicore processors.

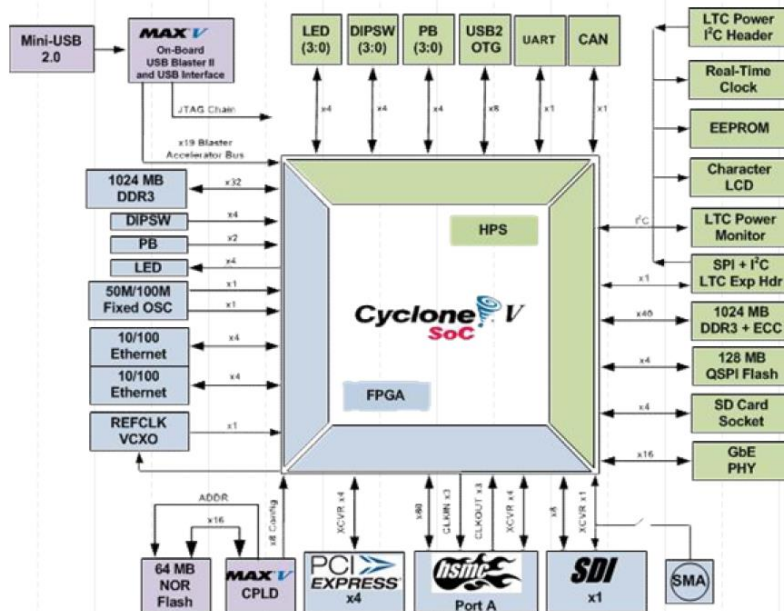
2

# Cyclone V SoC FPGA Device



3

# Cyclone V Peripheral Connections



4

# HPS

- HPS is a hard logic microprocessor unit (MPU) consisting of:
  - Dual-core ARM Cortex-A9 processor
  - On-chip memories
  - SDRAM
  - L3 interconnect
  - Support and interface peripherals
- The HPS will be used to execute the software portion of your SoC design

5

# FPGA

- FPGA component consists of:
  - FPGA fabric
  - Standard FPGA components (LUTs, CLBs, PLL etc.)
  - Shared memory controllers
  - General peripherals
- The FPGA is used to prototype hardware for your SoC design, receiving and sending data to and from the HPS using AXI buses, bridges, and Avalon master-slave devices

6

# Block Diagram of Intel Altera SoC FPGA

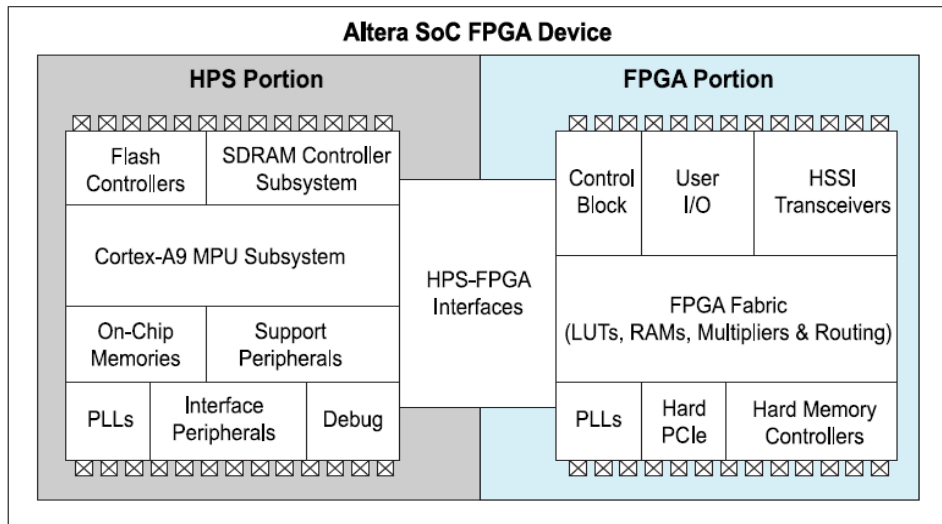


Figure 7-1. Altera SoC FPGA Device Block Diagram [2, pp. 1-1]

7

## HPS-FPGA Communication Interfaces: AXI Bridge

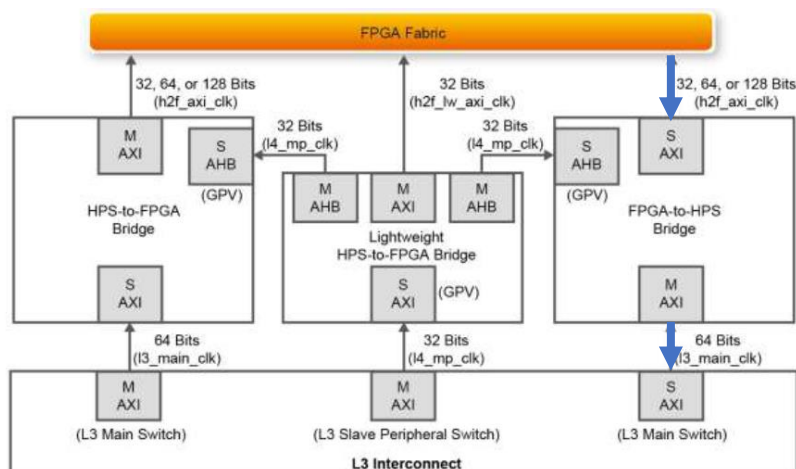


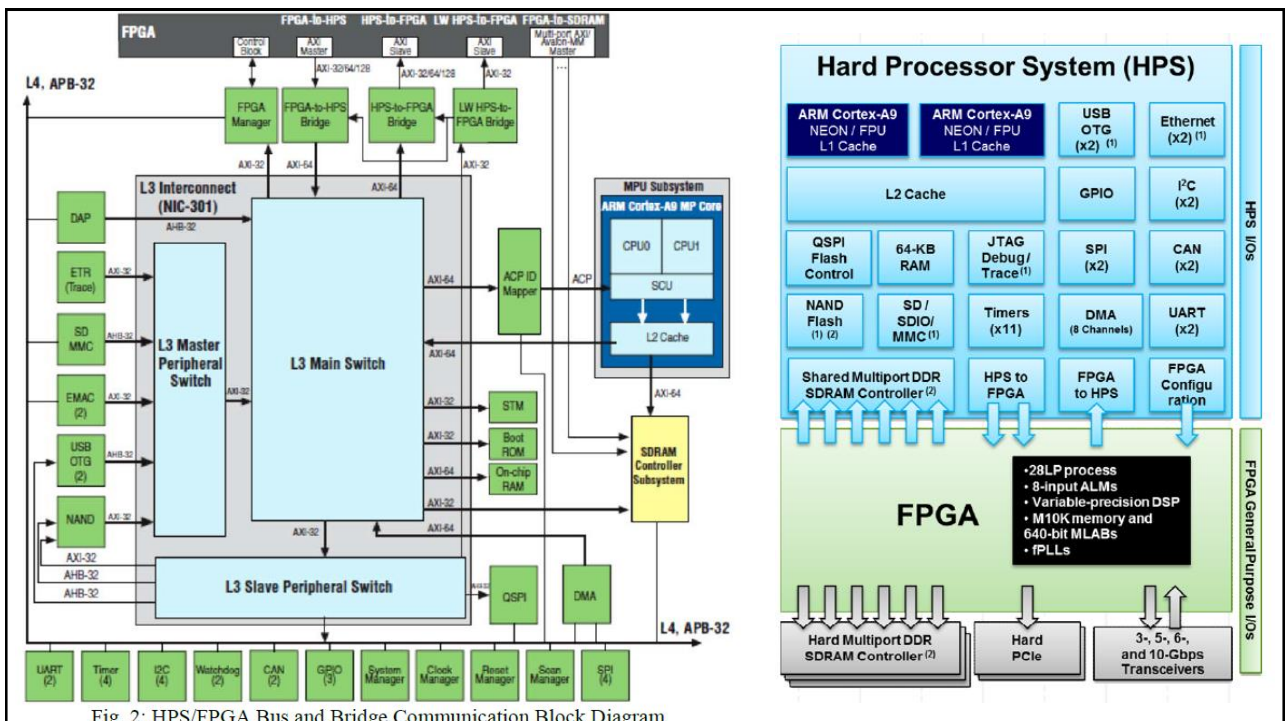
Figure 1-1 AXI Bridge Block Diagram

8

# HPS-FPGA Communication

- HPS logic and the FPGA fabric are connected through a series of **AXI (Advanced eXtensible Interface) Bridges**
- There are 3 main bridges used for communication:
  1. FPGA-to-HPS bridge (f2h)
  2. HPS-to-FPGA bridge (h2f)
  3. Lightweight HPS-to-FPGA bridge (**lwh2f**)
- Slaves are allowed to communicate back to the HPS through the FPGA-to-SDRAM connections provided by the FPGA's Avalon Memory Mapped (MM) Master
- Intel Altera system integration tool **Platform Designer** (previously called **Qsys**) is used to design the system and the communication between HPS and FPGA

9



10

## HPS-FPGA Communication

- HPS supports communication with the FPGA/peripherals through the L3 interconnect, which is connected to the HPS (DDR3) SDRAM Controller.
- Therefore, it is essential that SDRAM pins are configured correctly so that the HPS may read/write data to/from the SDRAM controller and establish communication between the L3 interconnect and FPGA.
- Once all hardware has been correctly prototyped, communication between the HPS and FPGA is programmed through a **memory mapped C application**.
- **Memory mapping** allows the CPU to view and access the FPGA's address space (containing our components) so that we may read/write information as necessary, controlling the hardware through software.
- The C application you will develop uses APIs to send write (or receive read) data to (and from) specified memory addresses.

11

## HPS-FPGA Communication

- Each of the IP components you add to your system possess a **base address**.
- You will use the base addresses to access, control, and send data to and from your SoC components using your C application.
- These addresses will be generated for you as header files using the NIOS II Command Shell.
- Common base addresses given in Table below:

Table I: Common Address Space Regions for Bridge Access

Region Name	Description	Base Address	Size
FPGA Slaves	For accessing FPGA slaves connected to the h2f bridge	0xC0000000	960MB
HPS Peripherals	Accessing slaves directly connected to the HPS	0xFC000000	64MB
Lightweight FPGA Slaves	Accessing slaves connected to the lwh2f bridge	0xFF200000	2MB

12

## General Design Flow

- Once your C application is complete, a binary is generated by compiling your software on a host computer.
- The binary must be placed on the microSD card.
  - This can be done also by first placing the binary on a USB drive, which will be inserted and mounted to the HPS/FPGA system. You must then copy the binary from the USB to your HPS home directory to execute the application.
  - Note: There are several ways to copy executables to the micro SD card; see separate tutorial on dejazzer.
- Upon execution, the HPS will communicate with the FPGA prototype based on the APIs and functionality you have coded in your C application.
- You may access and interact with the HPS/Linux OS from your host computer using a serial connection (Putty or minicom or the DS-5 terminal).

13

## General Design Flow

- There are many steps to follow for designing an HPS+FPGA SoC
- A structured block diagram outlining the general tools used for the hardware and software flows is shown below:

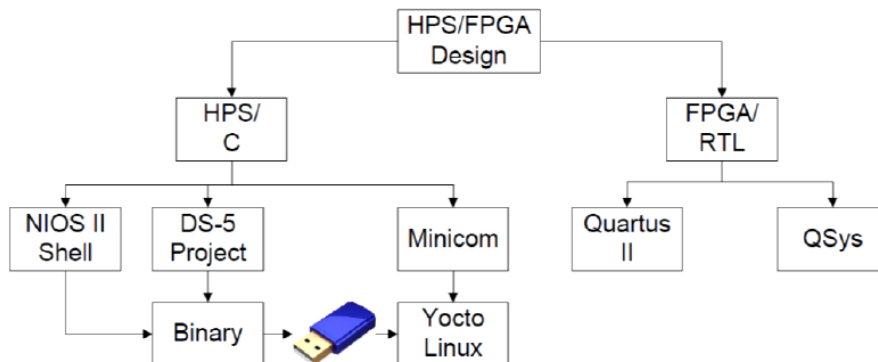
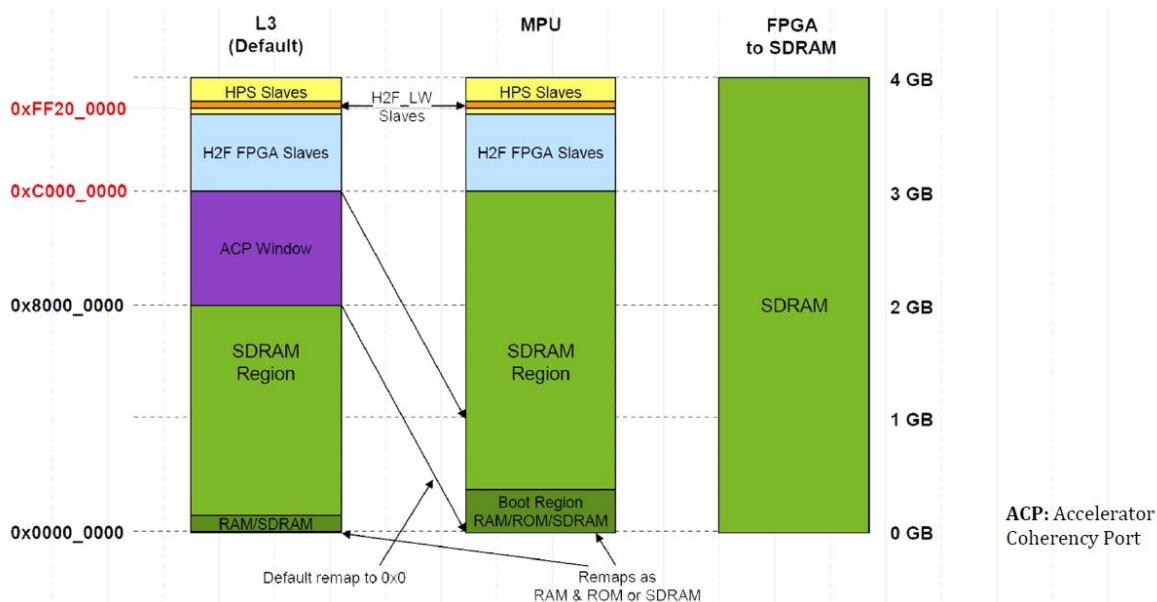


Fig. 4: Tools and Flow used for DE1-SoC Design

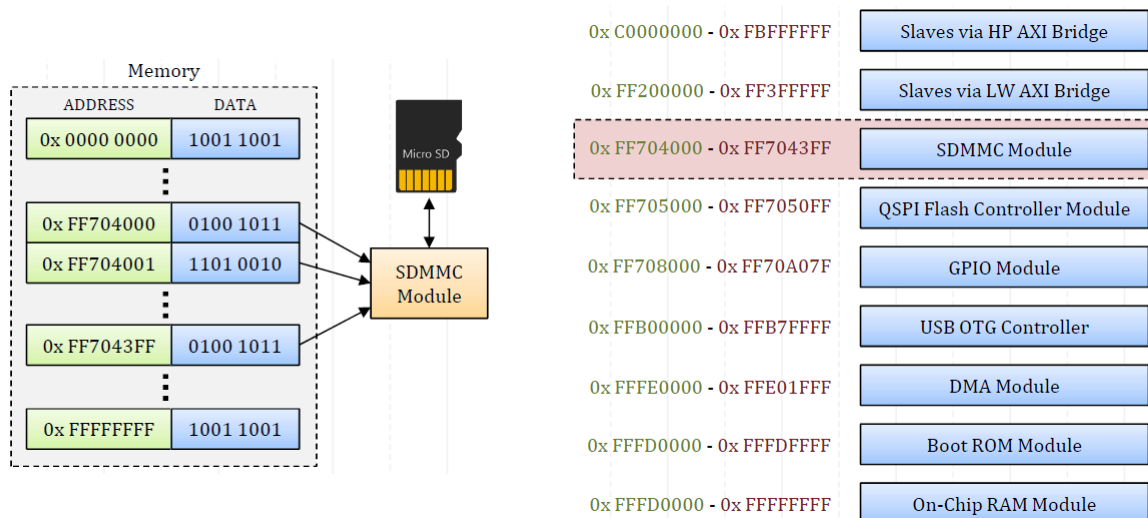
14

# Physical Address Mapping



15

# Cyclone V HPS Memory Map

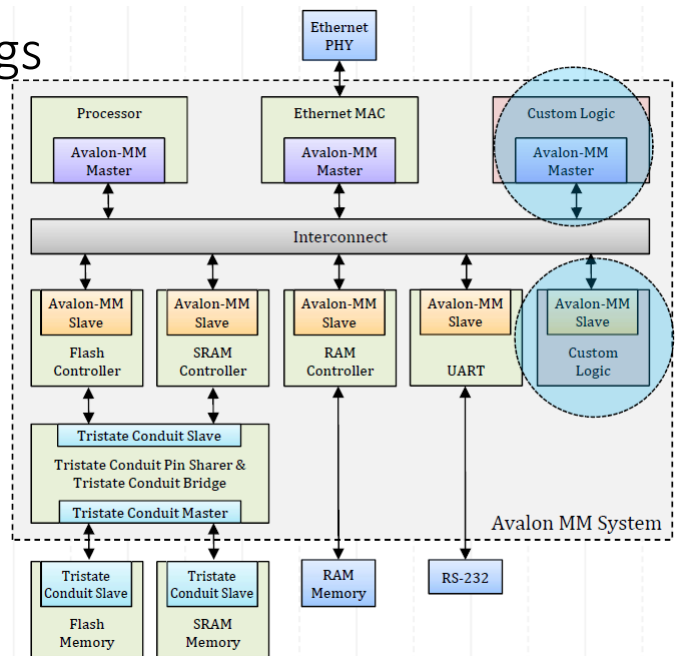


16



# Custom Logic Bindings

1. Custom Logic can be a master or slave
2. Usually NIOS CPU (Soft IP) is Avalon-MM Master
3. Usually any custom I/O is Avalon-MM Slave

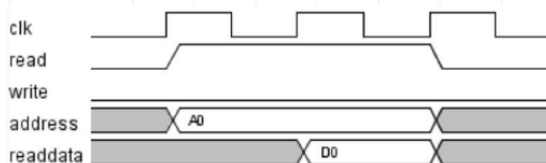


17

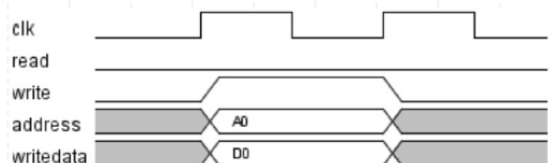
## Avalon-MM Slave Interface

Name	Width	Direction	Comments
avs_address	64 bits	Input	Address of slave being accessed
avs_read	1 bit	Input	Read operation requested
avs_write	1 bit	Input	Write operation requested
avs_readdata	8, 16, 32, or 64 bits	Output	Data read from slave
avs_writedata	8, 16, 32, or 64 bits	Input	Data to be written to slave

Read Waveforms:



Write Waveforms:

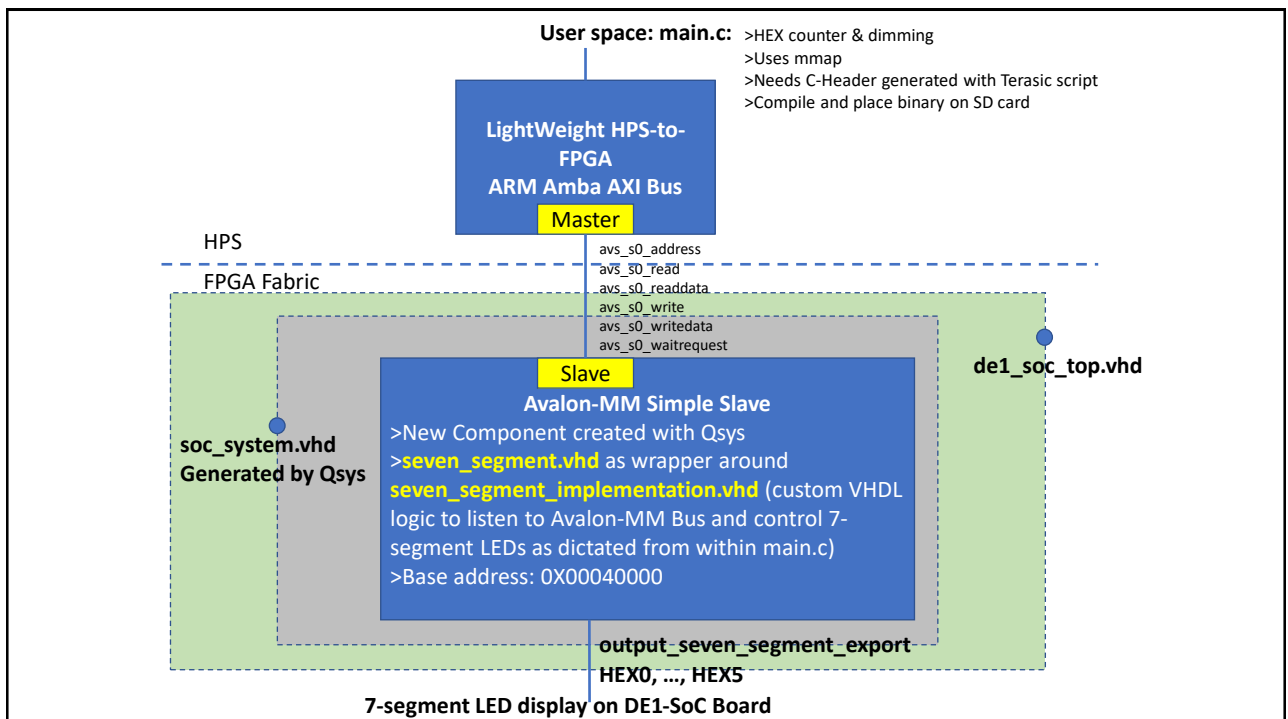


18

# References

1. Sahand Kashani-Akhavan, SoC FPGA Design Guide, DE1-SoC, <https://github.com/sahandKashani/SoC-FPGA-Design-Guide>
2. Systems-on-Chip Design COE838 / EE8221, Ryerson University, <https://www.ee.ryerson.ca/%7Ecourses/coe838/announcements.html>
3. ECE 5760, Advanced Microcontroller Design and system-on-chip, Cornell, <https://people.ece.cornell.edu/land/courses/ece5760>

19



20