Nios II 32 bit

**IS32, Computer Organization and Architecture**

**Report**

**on**

**Processor Architecture**

**Name : Pavan Krishna**

**USN: 1MS17IS051**

**INTRODUCTUON TO THE PROCESSOR**



Nios II is a 32-bit embedded-processor architecture designed specifically for the Altera family of FPGAs. Altera Corporation is an American manufacturer of programmable logic devices (PLDs), reconfigurable complex digital circuits. ... Altera and Intel announced on June 1, 2015 that they have agreed that Intel would acquire Altera in an all-cash transaction valued at approximately $16.7 billion.

Nios II incorporates many enhancements over the original Nios architecture, making it more suitable for a wider range of embedded computing applications, from DSP to system-control.

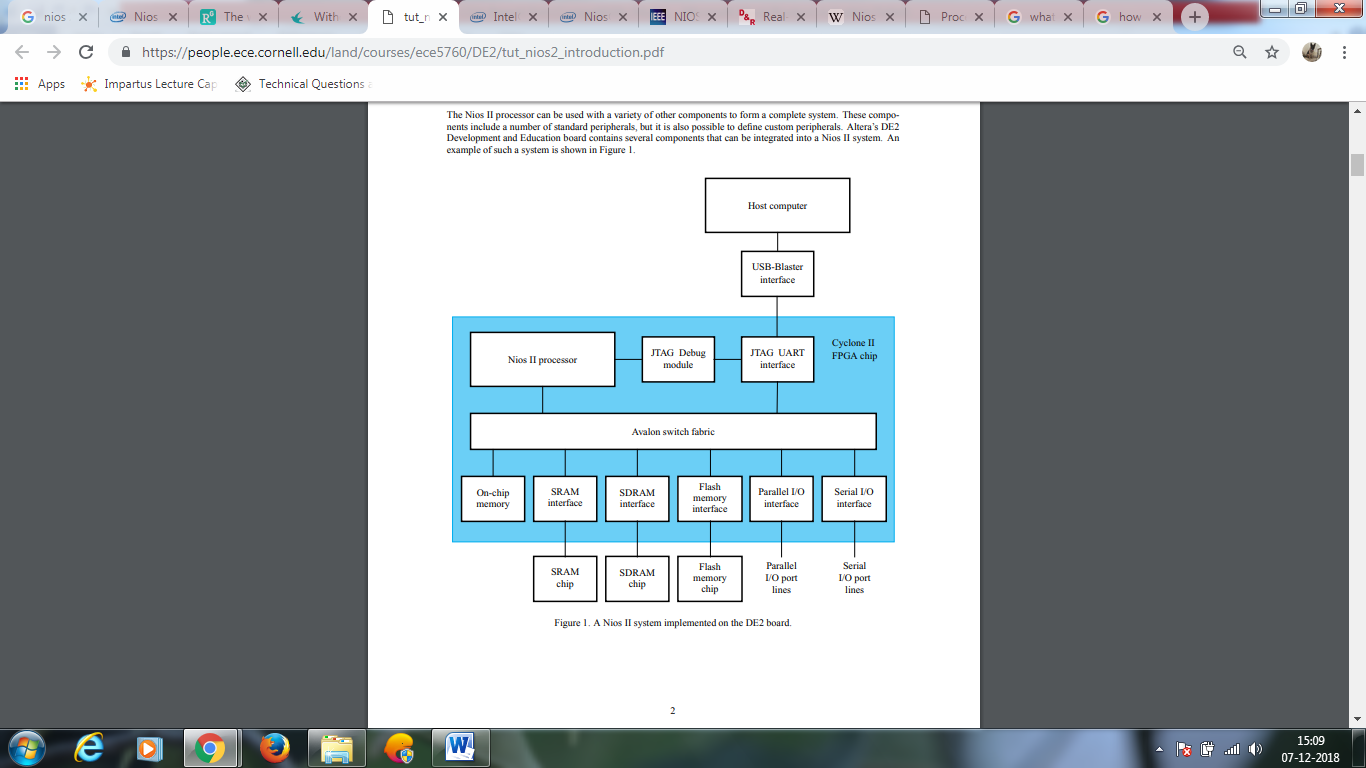
Like the original Nios, the Nios II architecture is a RISC soft-core architecture which is implemented entirely in the programmable logic and memory blocks of Altera FPGAs. The soft-core nature of the Nios II processor lets the system designer specify and generate a custom Nios II core, tailored for his or her specific application requirements. System designers can extend the Nios II's basic functionality by adding a predefined memory management unit, or defining custom instructions and custom peripherals.

**Nios II System**

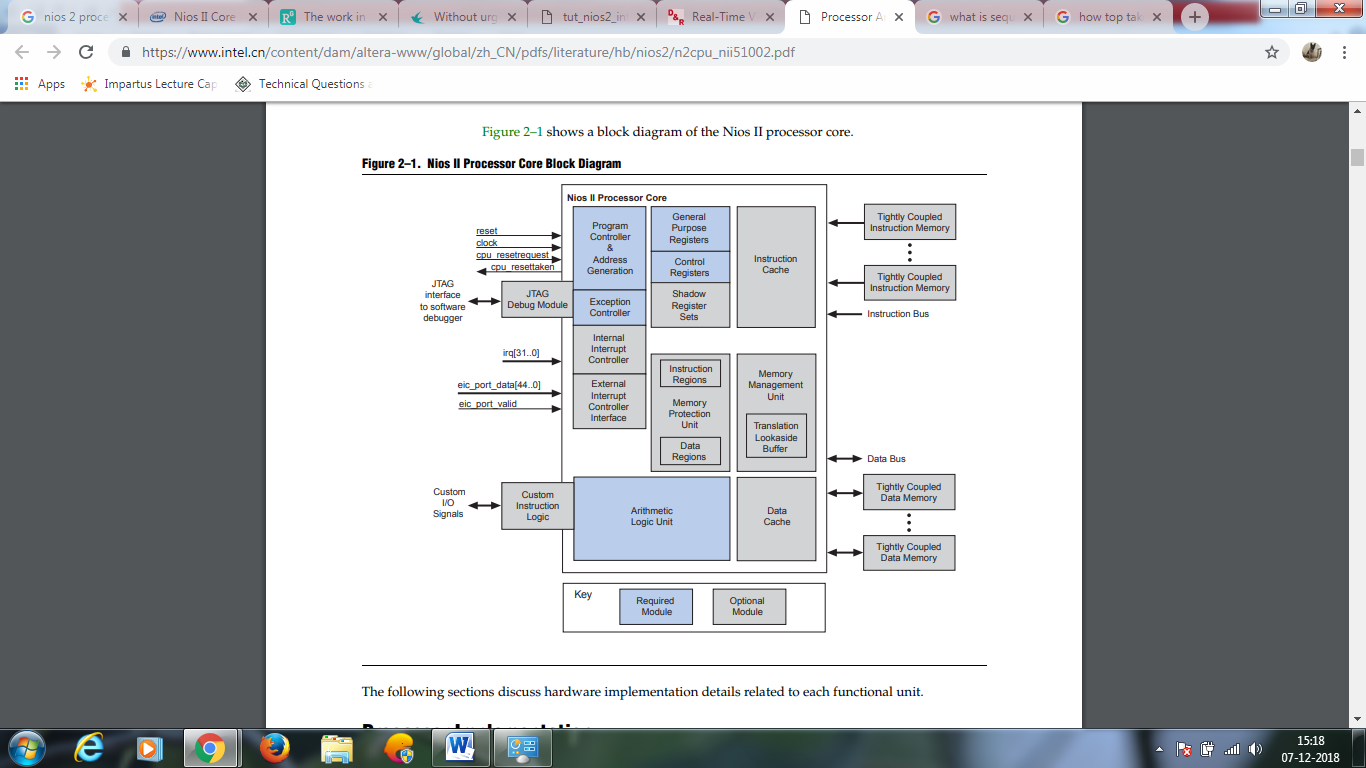
The Nios II processor can be used with a variety of other components to form a complete system. These components include a number of standard peripherals, but it is also possible to define custom peripherals. Altera’s DE2

Development and Education board contains several components that can be integrated into a Nios II system. An

example of such a system is shown in Figure.



**Nios II Processor Core Block Diagram**



**Processor Implementation**

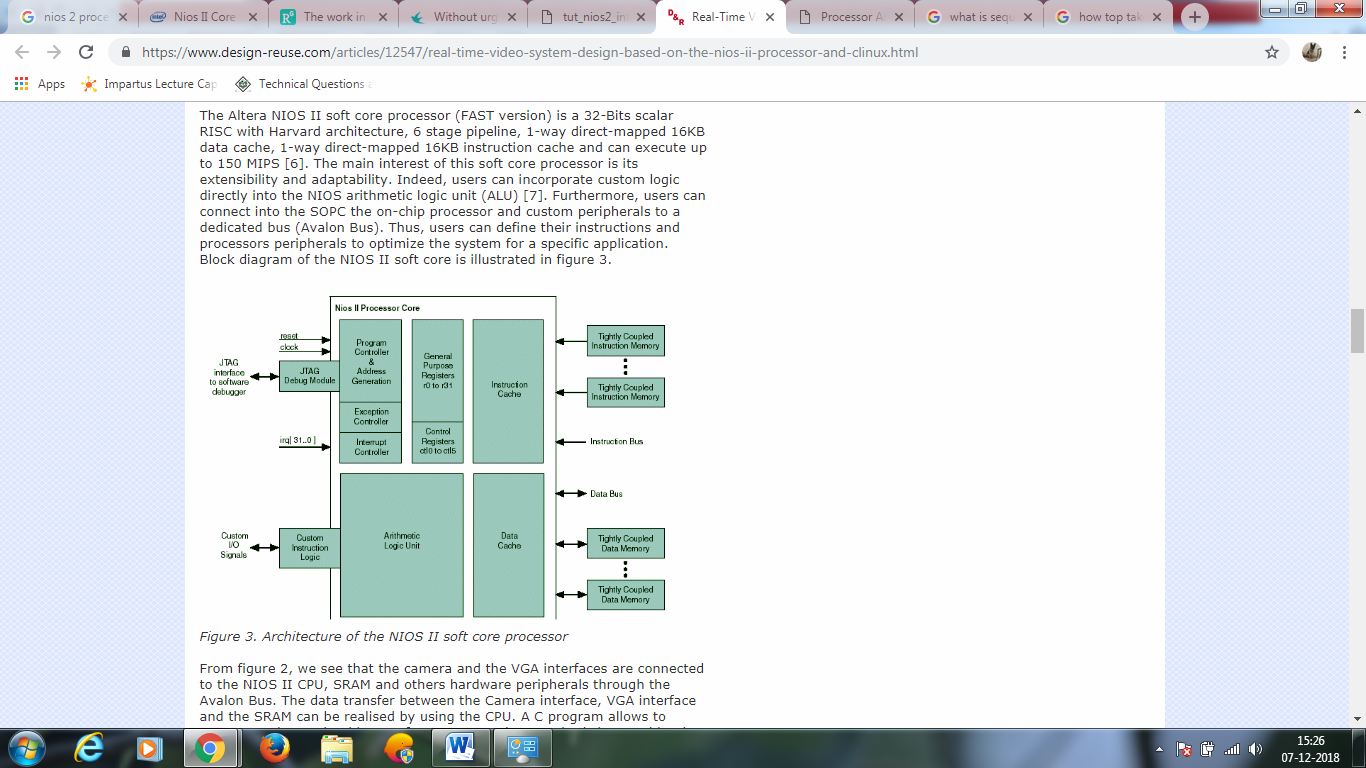
The functional units of the Nios II architecture form the foundation for the Nios II instruction set. However, this does not indicate that any unit is implemented in hardware. The Nios II architecture describes an instruction set, not a particular hardware implementation. A functional unit can be implemented in hardware, emulated in software, or omitted entirely. A Nios II implementation is a set of design choices embodied by a particular Nios II processor core. Each implementation achieves specific objectives, such as smaller core size or higher performance. This flexibility allows the Nios II architecture to adapt to different target applications.

Implementation variables generally fit one of three trade-off patterns: more or less of a feature; inclusion or exclusion of a feature; hardware implementation or software emulation of a feature. An example of each trade-off follows:

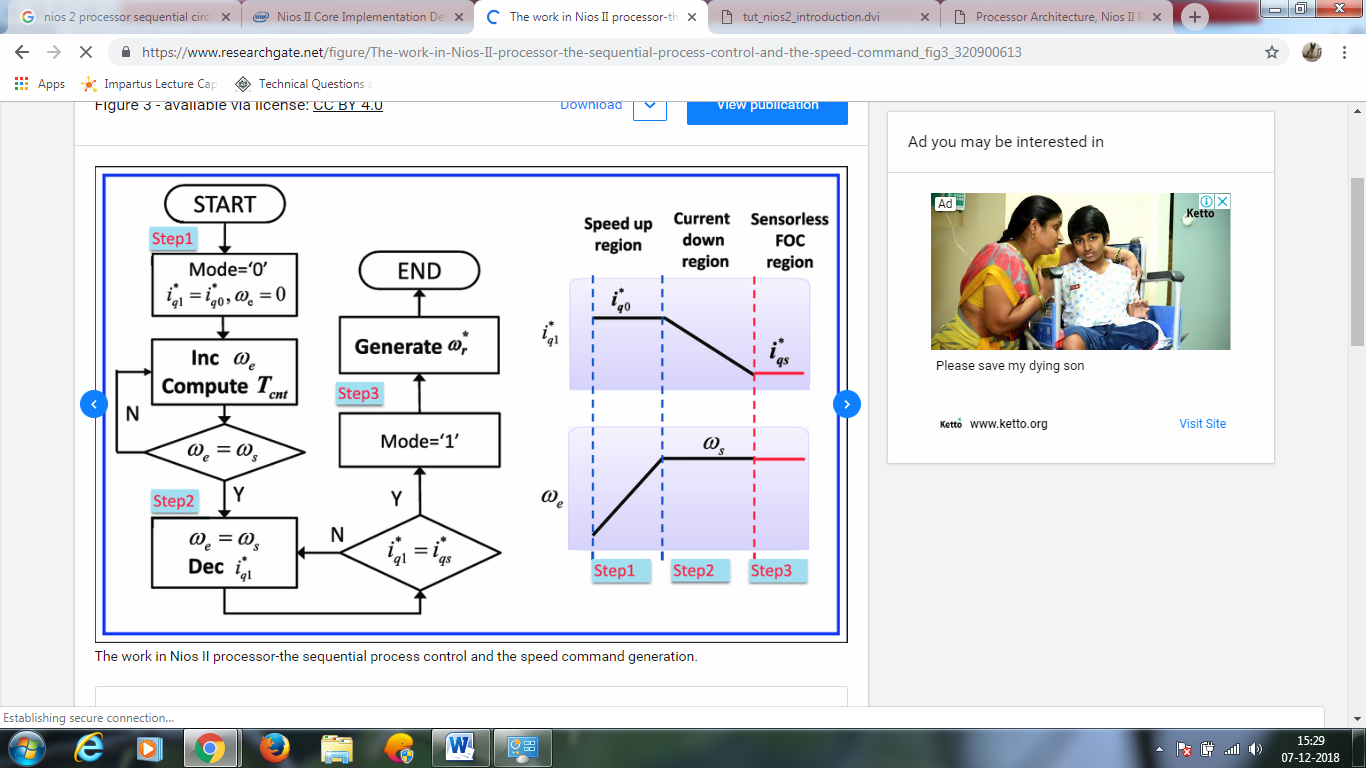
■ More or less of a feature—for example, to fine-tune performance, you can increase or decrease the amount of instruction cache memory. A larger cache increases execution speed of large programs, while a smaller cache conserves on-chip memory resources.

■ Inclusion or exclusion of a feature—For example, to reduce cost, you can choose to omit the JTAG debug module. This decision conserves on-chip logic and memory resources, but it eliminates the ability to use a software debugger to debug applications.

■ Hardware implementation or software emulation—For example, in control applications that rarely perform complex arithmetic, you can choose for the division instruction to be emulated in software. Removing the divide hardware conserves on-chip resources but increases the execution time of division operations.



**The work in Nios II processor-the sequential process control and the speed command generation.**



**Register Structure**

The Nios II processor has thirty two 32-bit general purpose registers, as shown in Figure . Some of these registers are intended for a specific purpose and have special names that are recognized by the Assembler.

• Register r0 is referred to as the zero register. It always contains the constant 0. Thus, reading this register returns the value 0, while writing to it has no effect.

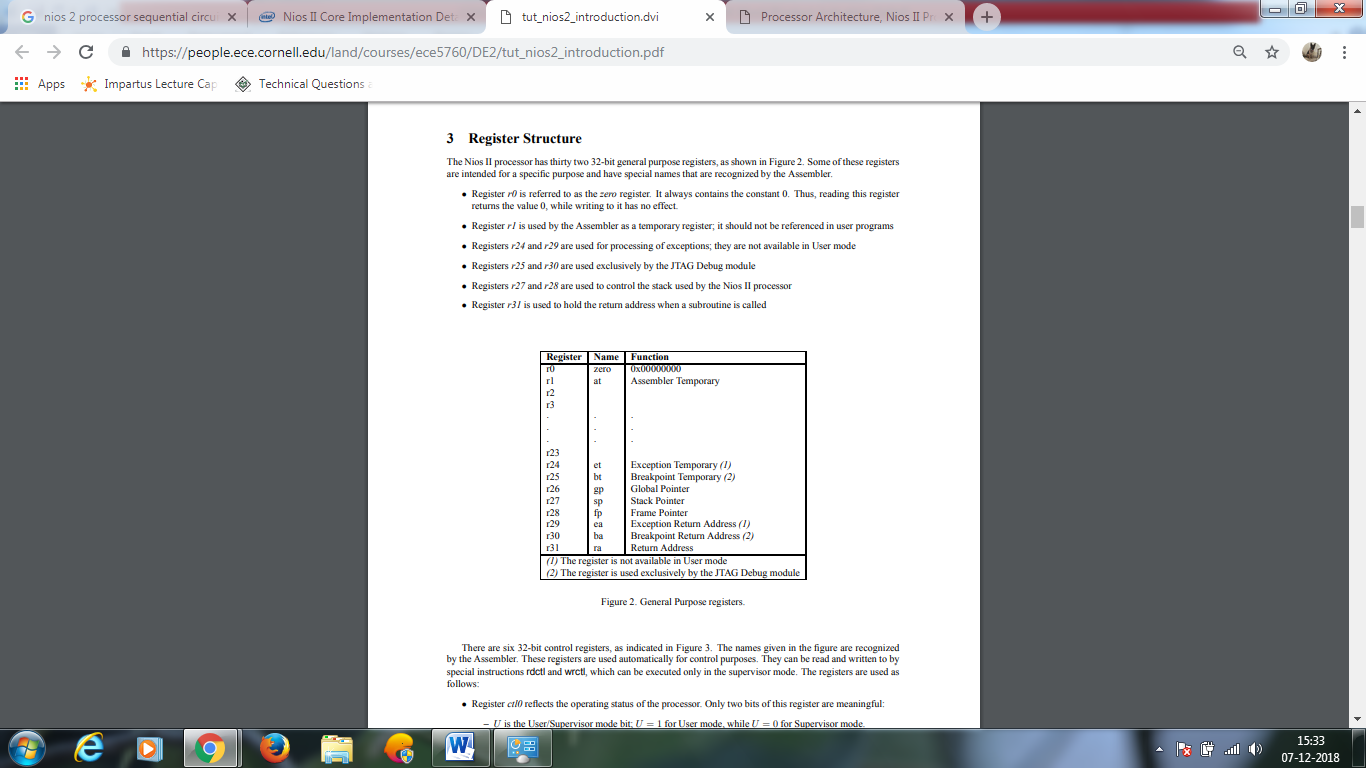
• Register r1 is used by the Assembler as a temporary register; it should not be referenced in user programs

• Registers r24 and r29 are used for processing of exceptions; they are not available in User mode

• Registers r25 and r30 are used exclusively by the JTAG Debug module

• Registers r27 and r28 are used to control the stack used by the Nios II processor

• Register r31 is used to hold the return address when a subroutine is called



The Nios II processor can optionally have one or more shadow register sets. A shadow register set is a complete set of Nios II general-purpose registers. When shadow register sets are implemented, the CRS field of the status register indicates which register set is currently in use. An instruction access to a general-purpose register uses whichever register set is active.

**Memory and I/O Organization**

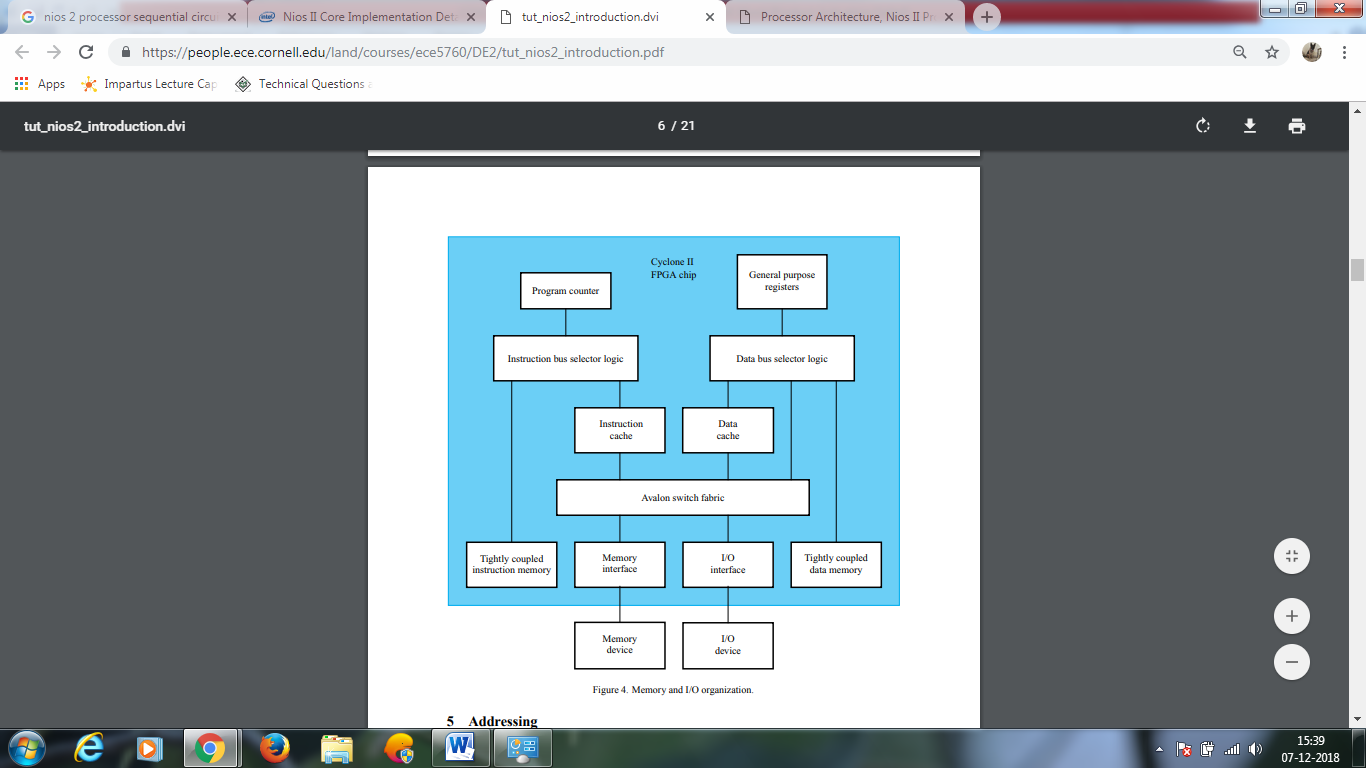
The flexible nature of the Nios II memory and I/O organization are the most notable difference between Nios II processor systems and traditional microcontrollers. Because Nios II processor systems are configurable, the memories and peripherals vary from system to system. As a result, the memory and I/O organization varies from system to system. A Nios II core uses one or more of the following to provide memory and I/O access:

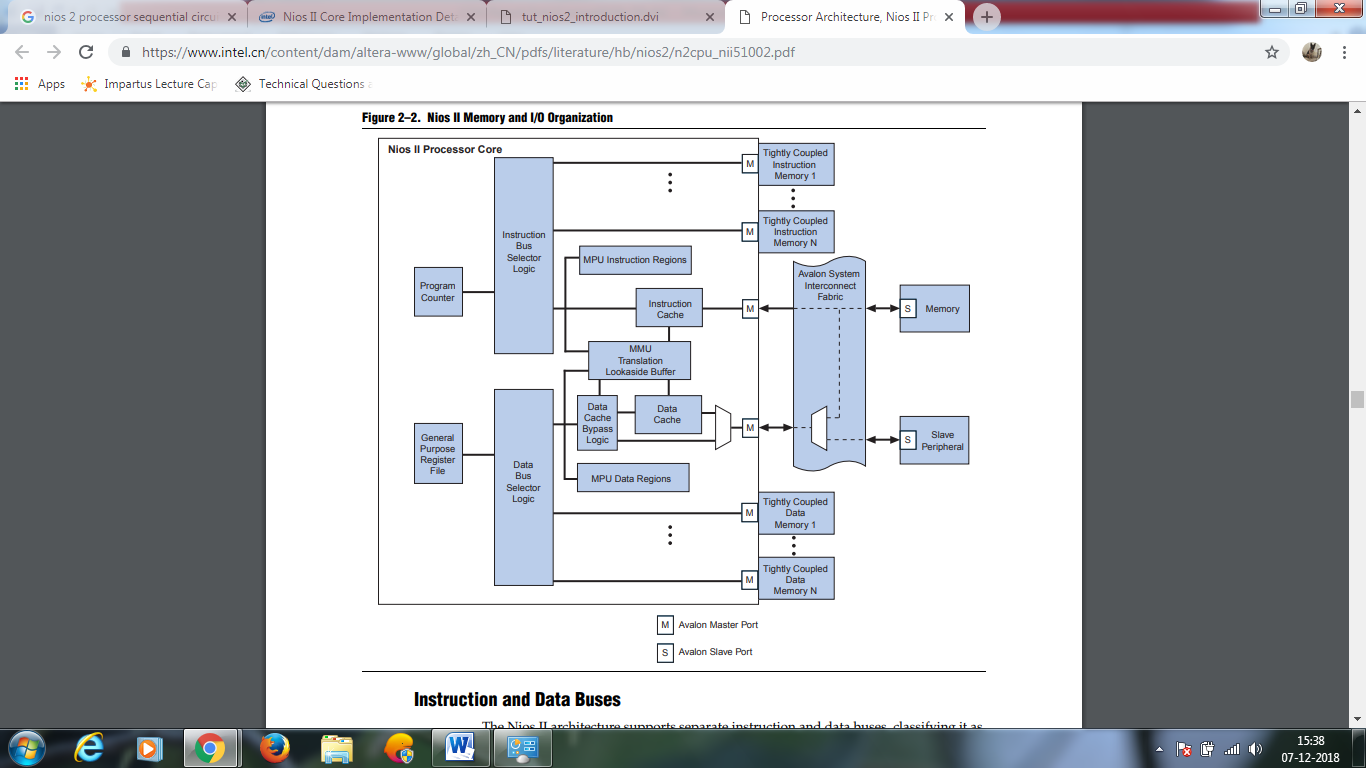
■ Instruction master port—An Avalon® Memory-Mapped (Avalon-MM) master port that connects to instruction memory via system interconnect fabric ■ Instruction cache—Fast cache memory internal to the Nios II core

■ Data master port—An Avalon-MM master port that connects to data memory and peripherals via system interconnect fabric

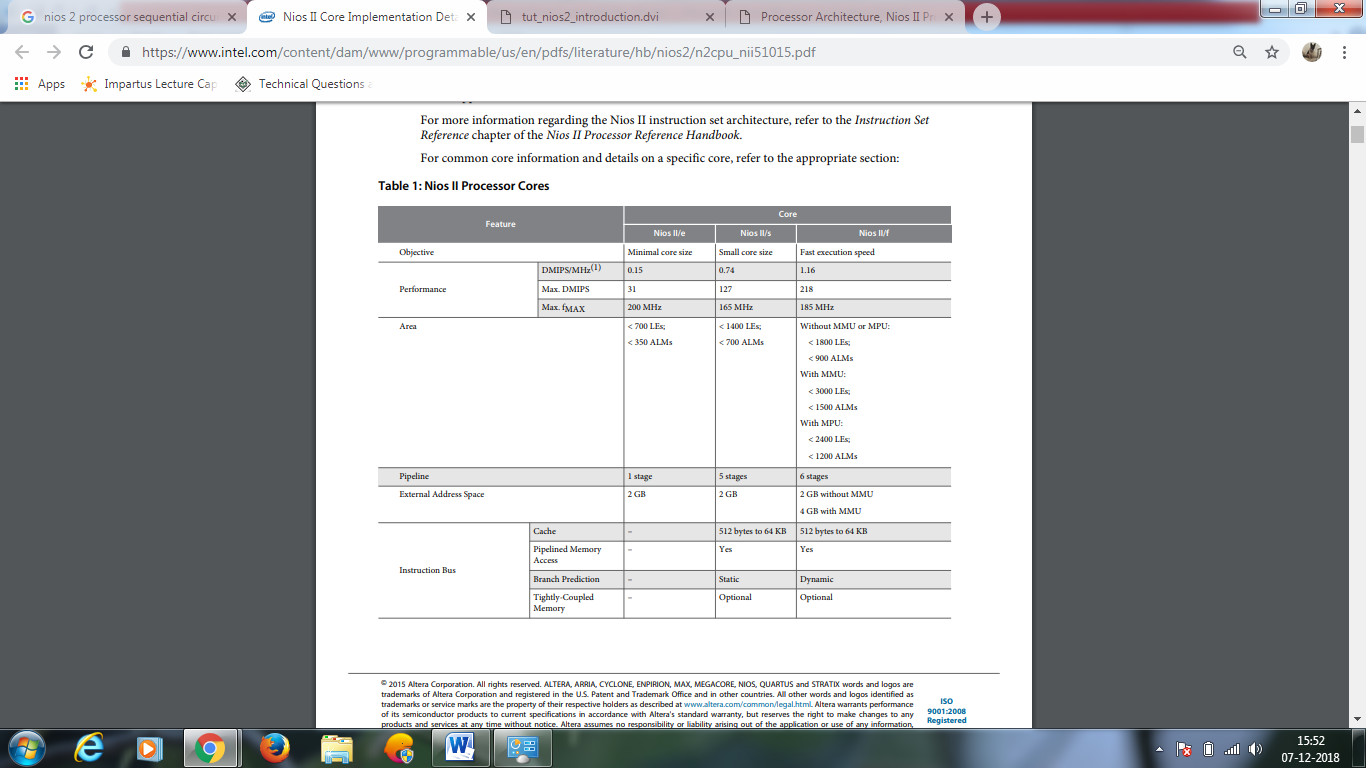
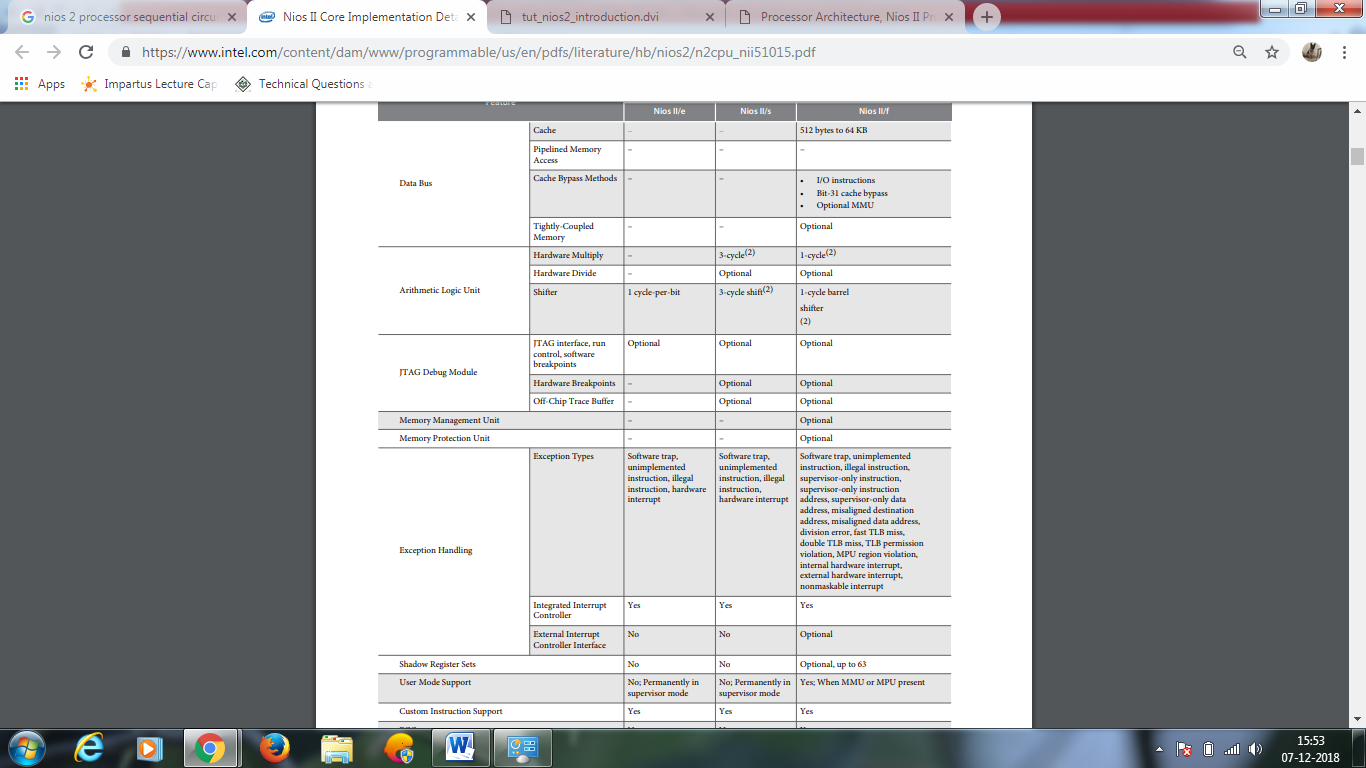
■ Data cache—Fast cache memory internal to the Nios II core

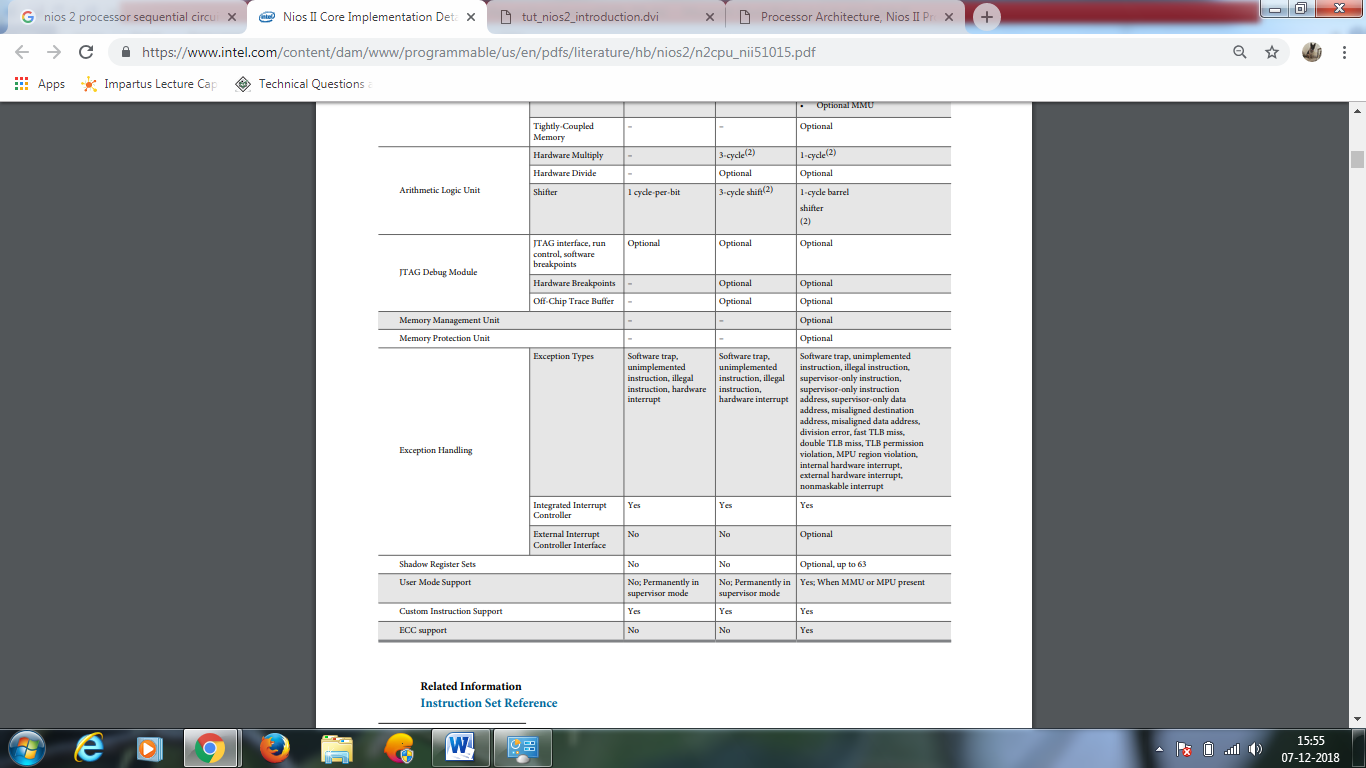
■ Tightly-coupled instruction or data memory port—Interface to fast on-chip memory outside the Nios II core The Nios II architecture handles the hardware details for the programmer, so programmers can develop Nios II applications without specific knowledge of the hardware implementation.





**Nios II Processor Cores**



**The Nios II/f core:**

• Has separate optional instruction and data caches

• Provides optional MMU to support operating systems that require an MMU

• Provides optional MPU to support operating systems and runtime environments that desire memory protection but do not need virtual memory management

• Can access up to 2 GB of external address space when no MMU is present and 4 GB when the MMU is present

• Supports optional external interrupt controller (EIC) interface to provide customizable interrupt prioritization

• Supports optional shadow register sets to improve interrupt latency

• Supports optional tightly-coupled memory for instructions and data

• Employs a 6-stage pipeline to achieve maximum DMIPS/MHz

• Performs dynamic branch prediction

• Provides optional hardware multiply, divide, and shift options to improve arithmetic performance

• Supports the addition of custom instructions

• Optional ECC support for internal RAM blocks (instruction cache, MMU TLB, and register file)

• Supports the JTAG debug module

• Supports optional JTAG debug module enhancements, including hardware breakpoints and real-time trace

### Nios II/s

Nios II/s core is designed to maintain a balance between performance and cost. Features of Nios II/s include:

* Instruction cache
* Up to 2 GB of external address space
* Optional tightly coupled memory for instructions
* Five-stage pipeline
* Static branch prediction
* Hardware multiply, divide, and shift options
* Up to 256 custom instructions
* [JTAG](about:blank) debug module
* Optional JTAG debug module enhancements, including hardware breakpoints, data triggers, and real-time trace

### Nios II/e

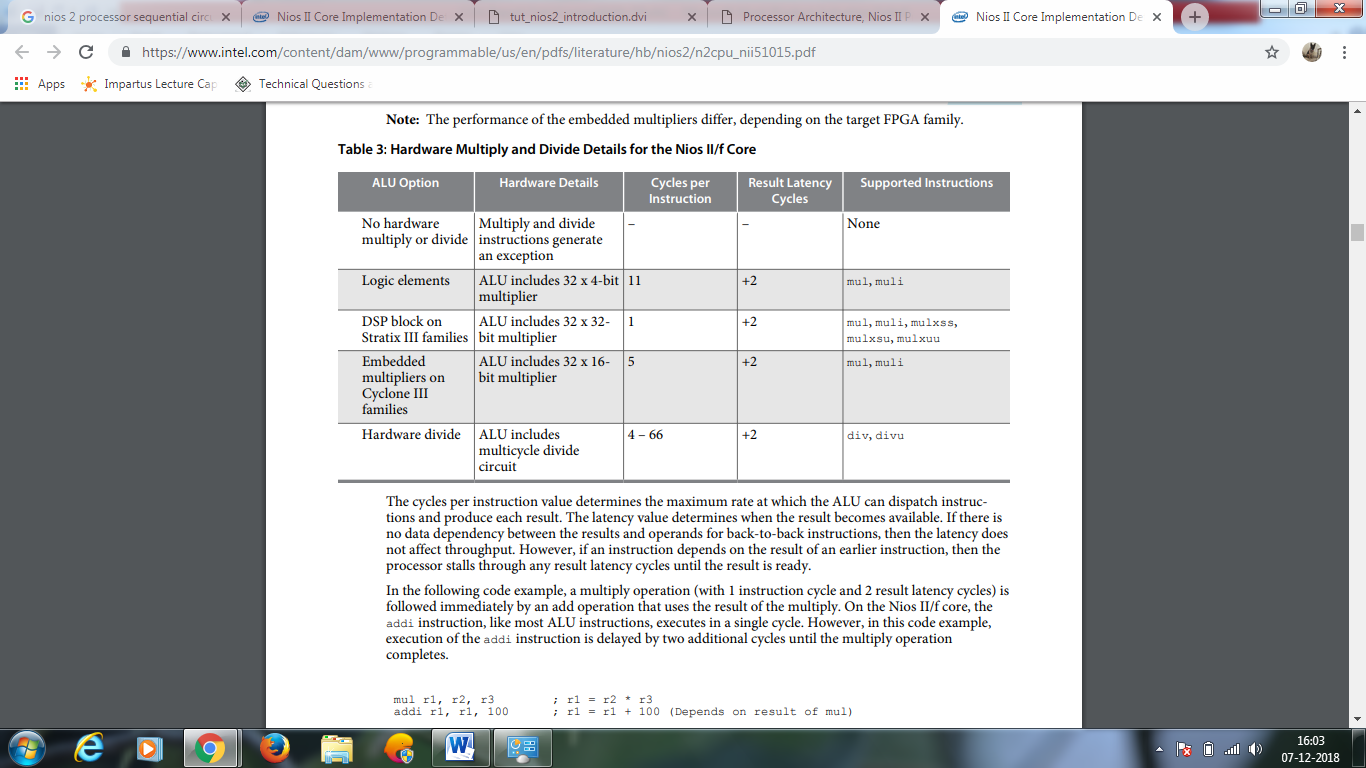
The Nios II/e core is designed for smallest possible logic utilization of FPGAs. This is especially efficient for low-cost Cyclone II FPGA applications. Features of Nios II/e include:

* Up to 2 GB of external address space
* [JTAG](about:blank) debug module
* Complete systems in fewer than 700 [LEs](about:blank#Architecture)
* Optional debug enhancements
* Up to 256 custom instructions
* Free, no license required

**Arithmetic Logic Unit**

The Nios II/f core provides several arithmetic logic unit (ALU) options to improve the performance of multiply, divide, and shift operations.

**Hardware Multiply and Divide Details for the Nios II/f Core**



**Overview of Nios II Processor Features**

The Nios II processor has a number of features that can be configured by the user to meet the demands of a desired system.

The processor can be implemented in three different configurations:

• Nios II/f is a "fast" version designed for superior performance. It has the widest scope of configuration options that can be used to optimize the processor for performance.

• Nios II/s is a "standard" version that requires less resources in an FPGA device as a trade-off for reduced performance.

• Nios II/e is an "economy" version which requires the least amount of FPGA resources, but also has the most limited set of user-configurable features.

The Nios II processor has a Reduced Instruction Set Computer (RISC) architecture. Its arithmetic and logic operations are performed on operands in the general purpose registers. The data is moved between the memory and these registers by means of Load and Store instructions. The wordlength of the Nios II processor is 32 bits. All registers are 32 bits long. Byte addresses in a 32-bit word can be assigned in either little-endian or big-endian style. The assignment style is one of the options that the user may select at configuration time. In this tutorial, we will use the little-endian assignment in which the lower byte addresses are used for the less significant bytes (the rightmost bytes) of the word. The Nios II architecture uses separate instruction and data buses, which is often referred to as the Harvard architecture.

A Nios II processor may operate in the following three modes:

• Supervisor mode – allows the processor to execute all instructions and perform all available functions. When the processor is reset, it enters this mode.

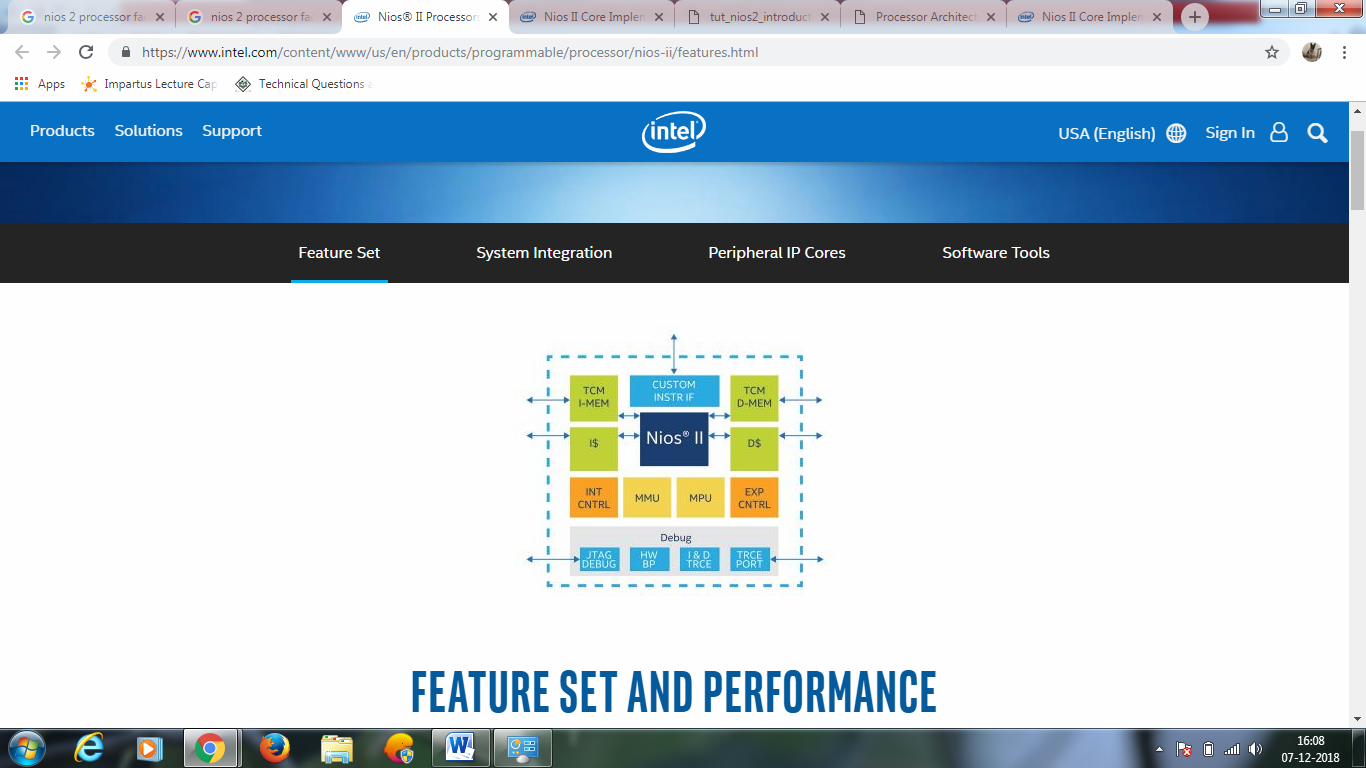
• User mode – the intent of this mode is to prevent execution of some instructions that shoud be used for systems purposes only. Some processor features are not accessible in this mode.

• Debug mode – is used by software debugging tools to implement features such as breakpoints and watchpoints. Application programs can be run in either the User or Supervisor modes.

Presently available versions of the Nios II processor do not support the User mode.

**Summary of Supported Features/Options**

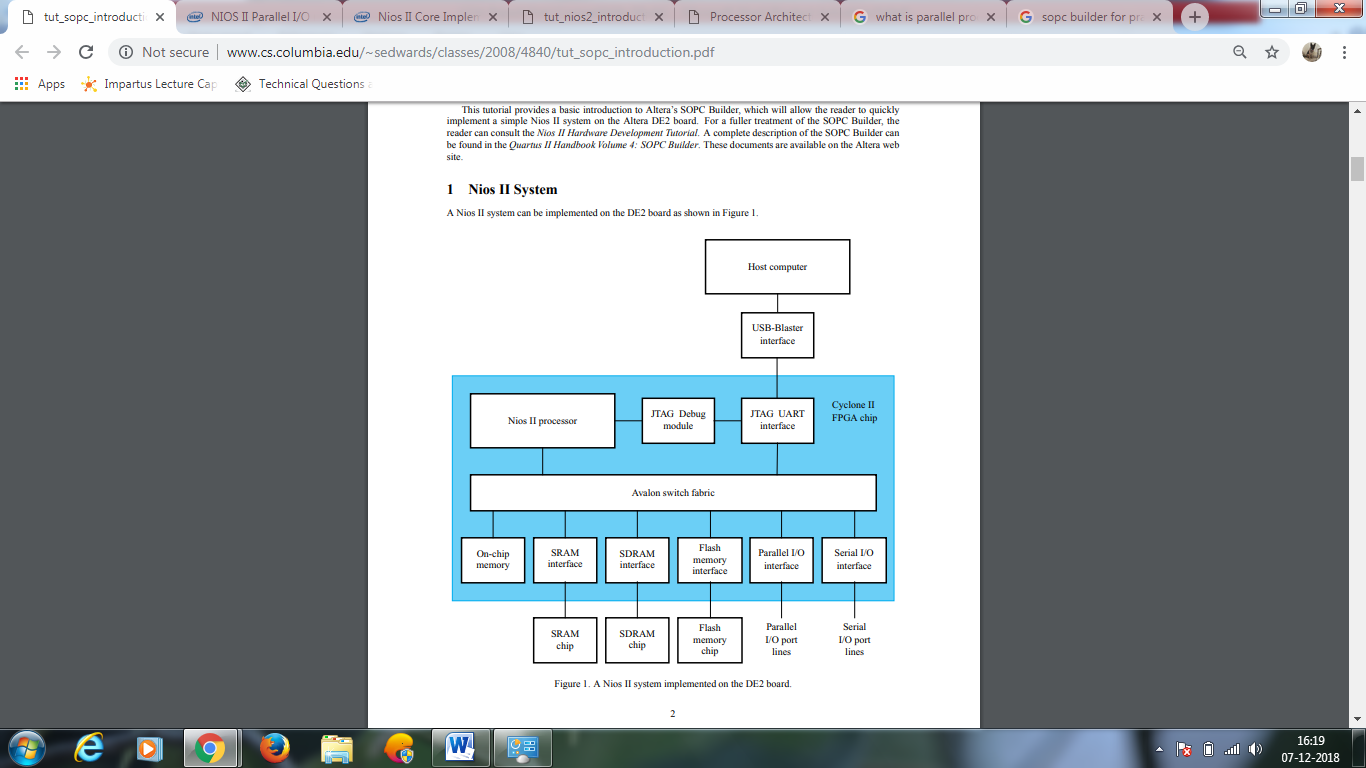
* MMU
* MPU
* External Vector Interrupt Controller with up to 32 interrupts per controller
* Advanced exception support
* Separate instruction and data caches (configurable from 512 bytes to 64 KB)
* Access to up to 4 GB of external address space
* Optional tightly-coupled memory for instructions and data
* Up to six-stage pipeline to achieve maximum DMIPS (Dhrystone 2.1 benchmark) per MHz
* Single-cycle hardware multiply and barrel shifter
* Hardware divide option
* Dynamic branch prediction
* Up to 256 [custom instructions](about:blank) and unlimited hardware accelerators
* Configurable JTAG debug module
* Optional JTAG debug module enhancements, including hardware breakpoints, data triggers, and real-time trace



**PRALLEL PROGRAMMING**

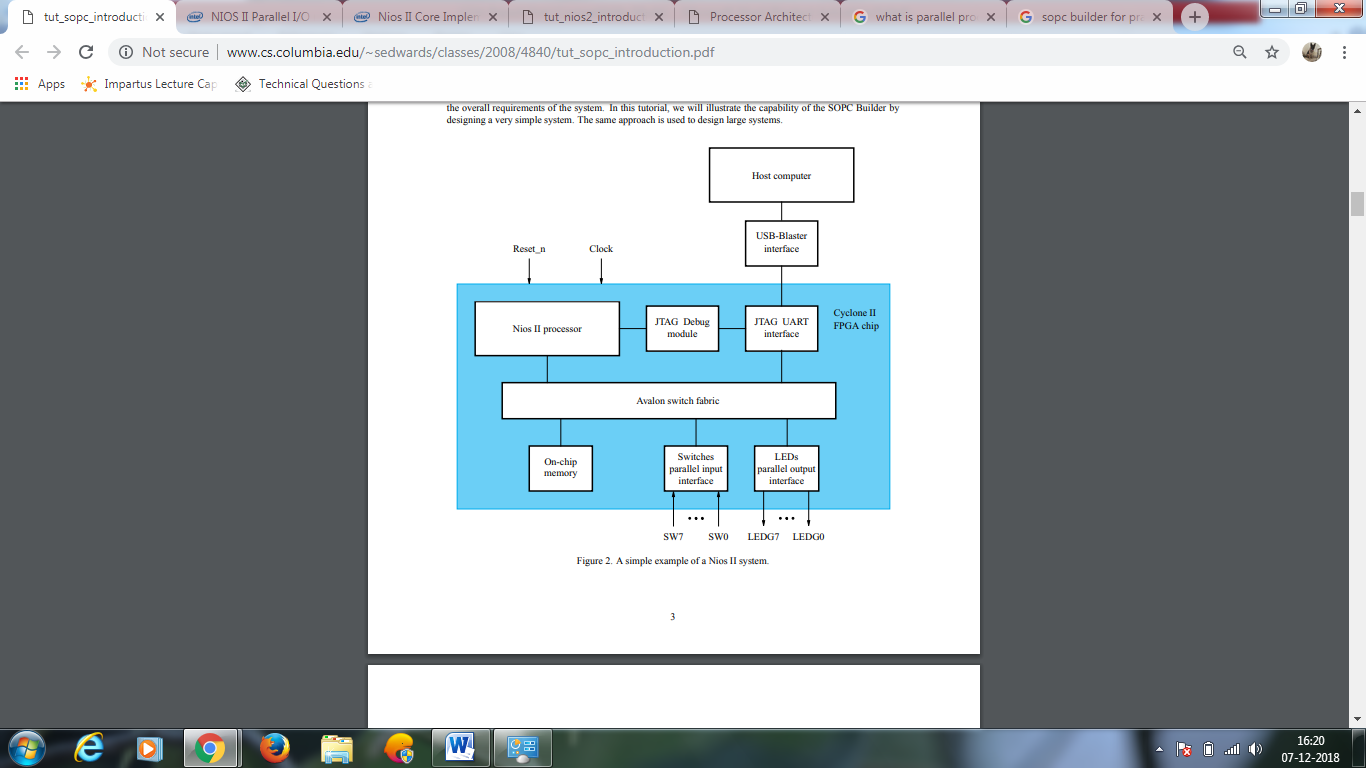
PPT at : [https://slideplayer.com/slide/8734632/](about:blank)

It Can be achieved through a SOPC Builder.



The Nios II processor and the interfaces needed to connect to other chips on the DE2 board are implemented in the Cyclone II FPGA chip. These components are interconnected by means of the interconnection network called the Avalon Switch Fabric. The memory blocks in the Cyclone II device can be used to provide an on-chip memory for the Nios II processor.

The SRAM, SDRAM and Flash memory chips on the DE2 board are accessed through the appropriate interfaces. Parallel and serial input/output interfaces provide typical I/O ports used in computer systems. A special JTAG UART interface is used to connect to the circuitry that provides a Universal Serial Bus (USB) link to the host computer to which the DE2 board is connected. This circuitry and the associated software is called the USB-Blaster. Another module, called the JTAG Debug module, is provided to allow the host computer to control the Nios II system. It makes it possible to perform operations such as downloading programs into memory, starting and stopping execution, setting breakpoints, and collecting real-time execution trace data. Since all parts of the Nios II system implemented on the FPGA chip are defined by using a hardware description language, a knowledgeable user could write such code to implement any part of the system. This would be an onnerous and time consuming task. Instead, one can use the SOPC Builder to implement a desired system simply by choosing the required components and specifying the parameters needed to make each component fit the overall requirements of the system. In this tutorial, we will illustrate the capability of the SOPC Builder by designing a very simple system. The same approach is used to design large systems.



## **External links, References and Sources**

[https://www.intel.com/content/www/us/en/products/programmable.html](about:blank)

[https://forums.intel.com/s/?language=en\_US](about:blank)

[https://web.archive.org/web/20090121133048/http://nioswiki.com/](about:blank)

[http://people.ece.cornell.edu/land/courses/ece5760/](about:blank)

[https://www.freertos.org/FreeRTOS-Nios2.html](about:blank)

[https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/nios2/n2cpu\_nii51002.pdf](about:blank)

[https://people.ece.cornell.edu/land/courses/ece5760/DE2/tut\_nios2\_introduction.pdf](about:blank)

[https://www.intel.in/content/www/in/en/products/programmable/processor/nios-ii/design-tools.html](about:blank)

[https://ieeexplore.ieee.org/document/4592663](about:blank)

[https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/nios2/n2cpu\_nii5v1.pdf](about:blank)