Digital Systems Track	Course Code: CND 221 ADVANCED FULL CUSTOM VLSI DESIGN	Project # 1

Supervisors:

Dr. Hatem Yousry

Dr. Mohamed Saleh

Project Title: 4-bit Microprocessor Design and Implementation

Project Description:

This project Aims to develop and implement 4 bits microprocessor. In this project, students will delve into the intricate aspects of microprocessor layout and circuit design. The project will emphasize hands-on experience in creating circuit designs, layout designs, and verifications for critical microprocessor components, such as the ALU, memory, control unit, and other circuitry using custom and standard cell libraries.

This project involves the schematic and layout design of a 4-bit microprocessor data path, including an ALU, a barrel shifter, and a register file using CMOS circuitry and Siemens VLSI design tools. The project will incorporate all the skills and knowledge students have gained from the individual lab assignments and provide a team of students with an opportunity to work on a more open-ended digital integrated circuits design problem.

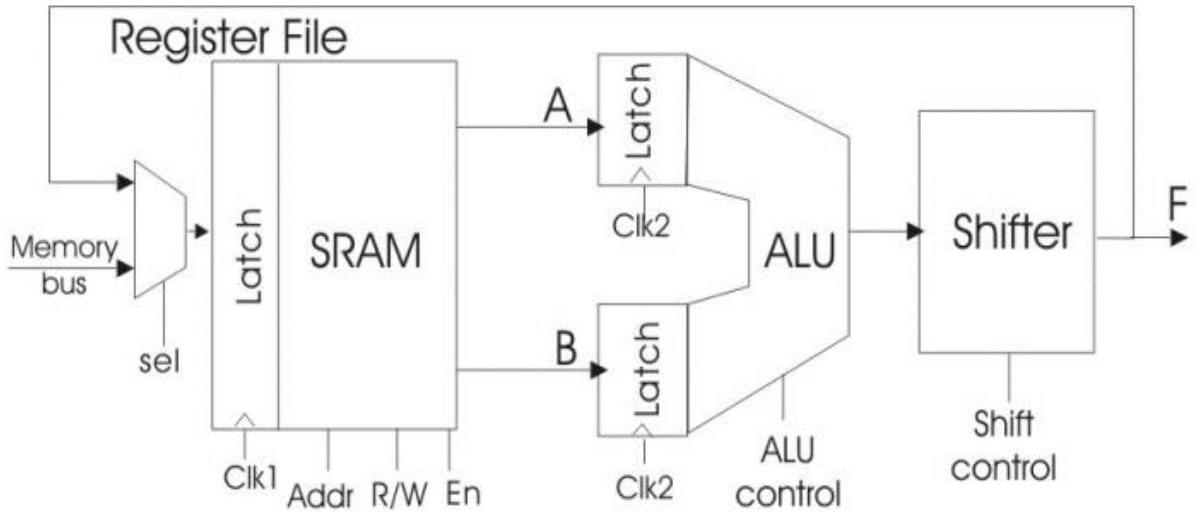

Figure 1. Structure of the microprocessor data path to be designed in this project.

DataPath:

The data path is the core element of a microprocessor that can perform many functions, including arithmetic and logic evaluation, data movement, and storage of results in a memory address specified by the instruction. In this design project, teams must design a complete 4-bit data path consisting of a register file, an ALU (Arithmetic Logic Unit) and a shifter, as shown in Figure 1. The data path will be capable of multiple instructions determined by several control signals.

The ALU can perform both arithmetic and bit-wise logical operations. The ALU will have two 4-bit data inputs and several control signals that specify the ALU function to be executed in a given clock cycle. The ALU input data is loaded into the latch at the rising edge of the ALU clock (clk2), and the output is generated after a delay due propagation through the combinational logic of the ALU and the shifter. The central component of an ALU is an adder, and for this project a Carry select adder will be used.

A shifter provides data manipulation capabilities. A barrel shifter will be used because it has a very efficient layout and can perform n-bit shifts in a single clock cycle.

The register file is a block of memory that provides the data inputs to and stores outputs from the ALU. The register file at read/write address is specified by control bits included in the opcode. In this project, the register file and ALU/shifter will be synchronized by two nonoverlapping clock phases that determine when data is latched into the register file (clk1) and into the ALU (clk2).

DataPath Design Specifications:

The following specifications must be achieved by the data path you design:

a. ***Bit width:*** 4
b. ***Inputs:*** one 4-bit word (from the memory bus), 13 control/address bits (6 for the ALU/shifter, 7 for the register file), two non-overlapping clocks (clk1, clk2).
c. ***Outputs:***
4-bit word (F), 1 carry-out
d. ***Functions***: must implement ***at least*** the following functions:
ALU functions:
- *A NOR B*
- *A XOR B*
- *A NAND B*
- *NOT A (invert)*
- *Increment A*
- *Decrement A*
- *A + B*
- *A * B (Bonus Function)*

If you implement only these functions, you must do so with only three ALU control signals. There is no explicit Cin bit; Cin must be generated from the ALU control bits.

Shifter: The shifter must implement:

- Pass (no shift)
- Shift left 1.
- Shift left 2.
- Shift right 1.
- Shift right 2.
- Rotate right 1.
- Rotate left 1.

Expanded functions

Two function expansion bits can be provided to implement additional ALU and/or shifter functions.

Register file

The 4x4 SRAM register file must implement 2-port read_from_memory and 1-port write_to_memory functions. Each clock cycle will be either a read or a write cycle, so two cycles will be needed to complete a function.

e. ***Power supply***: VDD = 3 volts, referenced to 0-volt ground.

f. ***Size specifications***: no specific layout size is required, but optimization of physical size is a design priority, and higher scores will be awarded to groups with smaller layouts. The floorplan in Fig. 2 will assist in achieving an optimal layout.

g. ***Timing specifications***: signal timing requirements are defined below. The worst-case propagation delay for all ALU + shifter functions must be optimized (e.g., less than **50ns**).

Control Bit Specifications:

It is recommended to complete this project with only the following 13 control signals: 1-bit register file input mux selection, two 2-bit register addresses, 1-bit register file read/write, 1-bit register file enable, 3-ALU controls, and 3-bits shift controls. These control signals must be given the signal names defined below. Note: this only defines the externally-supplied control signals; you can decode or otherwise manipulate them to construct any necessary internal control signals.

Data Path Layout:

The 4-bit datapath can be implemented by constructing a 1-bit slice horizontally and stacking 4 one-bit slices vertically. This layout floorplan for the data path is shown in Fig. 2. The data flows horizontally from the left to the right. The control signals run vertically across each slice. You may implement some functional blocks as 4 bits, but make sure you organize it in 1-bit slices that can be pitch-matched to the other functional blocks.

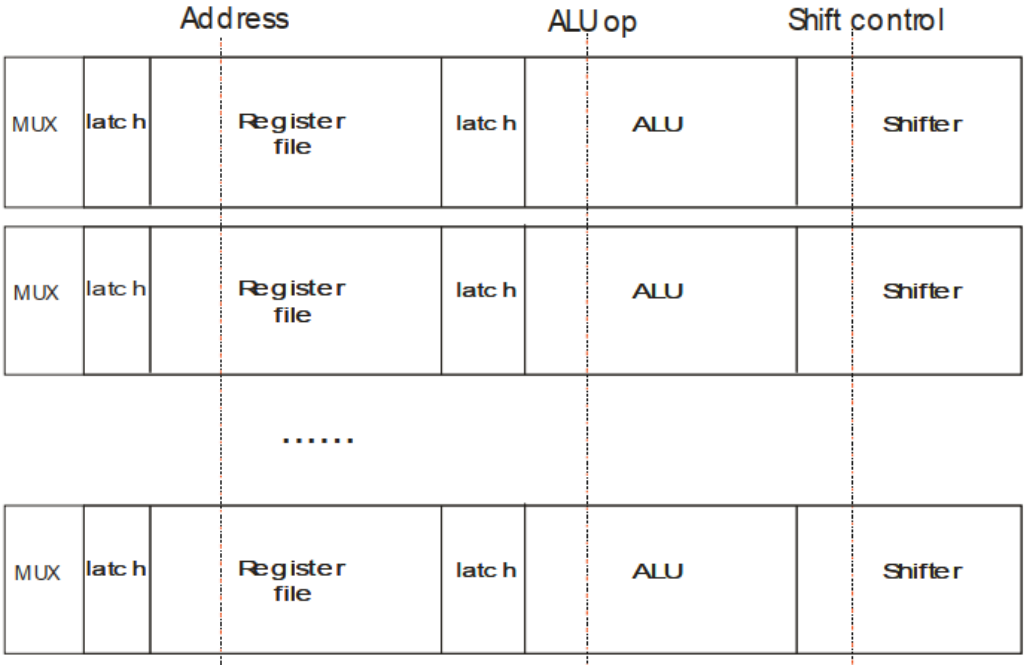

Figure 2: Floorplan of the datapath

Register File Description:

The 4-bit register file should be implemented as a 4x4 SRAM. Since the SRAM is small, you do not have to implement a sense amplifier. The register file should have two 4-bit read (output) ports and one 4-bit write (input) port. Input to the register file can come either from the ALU output for from an external memory bus, as selected by a MUX at the input of the register file. The memory bus should be implemented as a 4-bit input word. A 4-bit latch should be included to store the register file input word, which, during a write cycle, will be saved to the SRAM block.

The register file uses an enable signal for the general task of enabling inputs to and outputs from the register file at the proper time. The register file must be carefully designed to properly implement the enable function. In the read cycle, register file output is passed to the ALU when enable is high; otherwise, that output should be at high impedance. The output should also be at high impedance during the entire write cycle, when the R/W signal is high. In the write cycle, data is stored in the SRAM only when enable is high; otherwise, the data from the register file latch should not be allowed to affect the SRAM. To accomplish this, the enable signal should act on the SRAM address decoder outputs. The enable function allows the data path to implement functions that do not act on the register file, such as branches or status register checks, although these functions are not implemented in this design project.

The register file address contains two independent 2-bit addresses. Address "A" should specify the address for data passed to ALU input A, and address "B" should specify ALU input B. Address "A" is also used to specify the address for storing data to the SRAM during a write cycle. The timing of the data path should allow the value of address A to be changed before the write cycle, as shown in Fig. 3. However, for this project you can keep the address constant and store the ALU/shifter data back to the same location where input A was taken from.

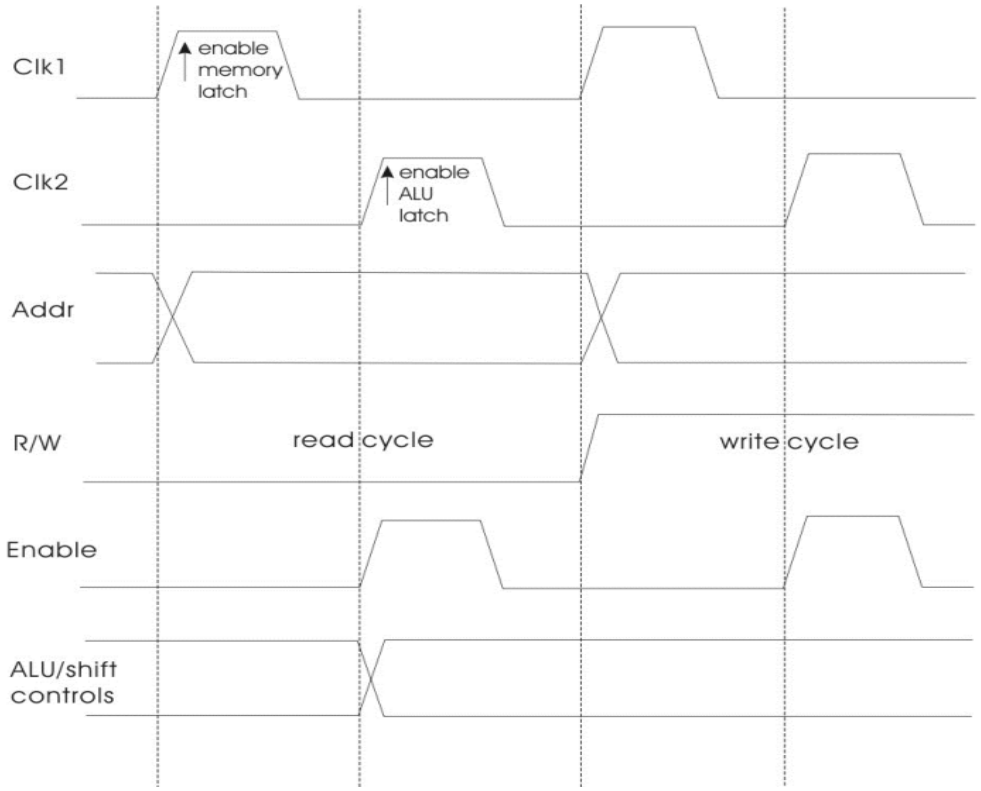

Figure 3: Timing diagram for a complete 2-cycle data path function. The first cycle is a register file.

Clock/Timing Description:

Two non-overlapping clock phases are required for the data path, as shown in Fig. 3. Data from the ALU/shifter or memory is loaded in the register file latch when clk1 is high and held there while clk1 is low. Data from the register file is loaded in the ALU latch when clk2 is high and held there when clk2 is low. Two cycles of the clocks are needed to complete a data path function; data is passed to and through the ALU/shifter in the "read" cycle, and ALU/shifter data is stored in the register file on the "write" cycle. Address and control signals for all data path functions must be synchronized to the clock signals specifically as shown in Fig. 3.

Project Requirements (Deliverables):

The following cell views for all circuit blocks must be created using the available tools:

- Schematic, Symbol, and Layout
- DRC, LVS, and Parasitic extraction.
- Circuit simulation (functional simulation of all datapath functions)
- Post layout simulation (functional simulation of all datapath functions after adding PEX file)

Input and output naming:

- **Inputs**
  - md<0:3>, memory bus data
  - rfs, reg. file mux selection
  - rfen, reg. file enable
  - rw, read/write
  - ada<0:1>, register address A
  - adb<0:1>, register address B
  - f<0:2>, shifter controls
  - f<3:5>, ALU controls
  - clk1, clk2
- **outputs:**
  - Y<0:3>
  - Cout

The following characteristics must be measured and reported:

- Slowest logic function and propagation delay for that function (ALU)
- Slowest arithmetic function and propagation delay from that function (ALU)
- Slowest propagation delay of the data path (ALU + Shifter)

  Hint: Propagation delays are measured from clock edge to last output transition.
- Physical area of the complete data path layout
- Total number of transistors in the data path (available from the final LVS output)
- Total, static, and dynamic power consumption for a full read and write cycle during an ADD and Shift_Left_2 operations.

Circuit Requirements:

- A Carry select adder must be implemented.
- A barrel shifter must be implemented.
- The register file memory must be implemented with a multi-port 4x4 SRAM.
- Minimum sized transistors can be used, but are not required.
- All required functions must be implemented using only the specified control/select inputs.

- You must employ hierarchical design where you instantiate lower-level cells into higher level circuit block, both in schematic and layout design
- Buffers should be designed and included wherever fan-out is larger than 5 gates.

Hint:

It is always best to pass DRC and LVS on smaller cells BEFORE using them to construct larger cells. Final LVS can be very time consuming for many groups.

Number of Students:

Each group should consist of 3-5 students.

Assessment Criteria:

- Circuit Design: 35%
  - ALU: 15%
  - Shifter: 5%
  - SRAM: 10%
  - Others: 5%
- Layout Design: 35%
  - ALU: 15%
  - Shifter: 5%
  - SRAM: 10%
  - Others: 5%
- System Integration + reporting the required project document, and presenting the results: 30%
- Bonus 10% (Do one of the following):
  - Apply/improve pipelining (e.g., Wallace tree)
  - Do the multiplication function (A * B)
  - Perform analysis including sizing and interconnect modeling for accurate delay, power consumption, and noise margin for the Register File.

Best of Luck!

Digital Systems Track	Course Code: CND 221 ADVANCED FULL CUSTOM VLSI DESIGN	Project #2

Supervisor:

Dr. Esraa Swillam

Project Title:

Simplified-Data Encryption standard (S-DES)

Project Description:

Data Encryption Standard (DES) was developed by IBM cryptography researcher Horst Feistel and is based on the Feistel block cipher.

Security is crucial in all data systems. Initially, the need for secure communication was driven by military and national security. Now, it's important in business and personal sectors too, with e-commerce necessitating secure internet communication. Many businesses use firewalls to safeguard their information, and personal privacy is a growing concern. Encryption is one way to secure communication, transforming data into an unrecognizable form that can only be decrypted by the intended recipient. The strength of an encryption scheme depends on its decryption process. Many communication products use encryption for security. This project discusses the Data Encryption Standard (DES), a well-known encryption algorithm published by NIST (National Institute of Standard and Technology). DES was introduced in 1976 and has a short key length of 56 bits which makes it insecure for nowadays complex needs, however it has been influential in advancing cryptography in general.

Description of the system using DES encryption is shown in figure1.

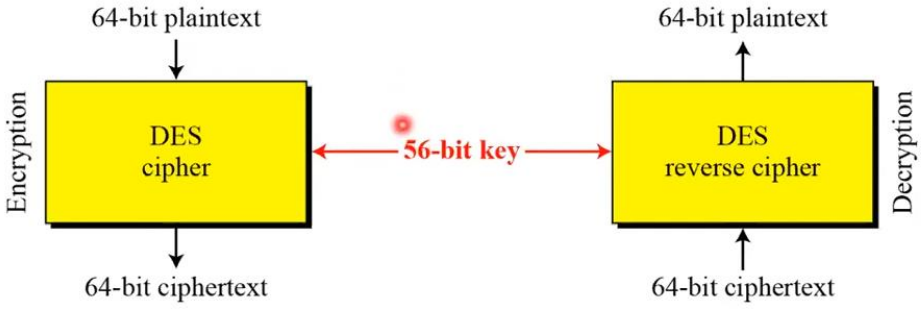

Figure 1 Encryption and Decryption with DES

Students are required to research and understand what is inside the DES cipher block. Here are the major steps in DES:

- Initial permutation and inverse permutation
- Subkey generation
- Feistel function
- Expansion and substitution
- S boxes

The scope of this project is to implement a Simplified DES (SDES) which has similar properties to DES but with much smaller parameters.

The S-DES encryption algorithm shown in Figure2 takes:

• 8-bit block of plaintext
• 10-bit key as input
• Produces an 8-bit block of ciphertext as output

The encryption algorithm involves five functions:

• An initial permutation (IP);
• A complex function labeled f_K, which involves both permutation and substitution operations and depends on a key input.
• A simple permutation function that switches (SW) the two halves of the data;
• The function fK again.
• A permutation function that is the inverse of the initial permutation (IP–1).
• We can concisely express the encryption algorithm as a composition of functions:

$$IP^{-1} \circ f_{K_2} \circ SW \circ f_{K_1} \circ IP$$

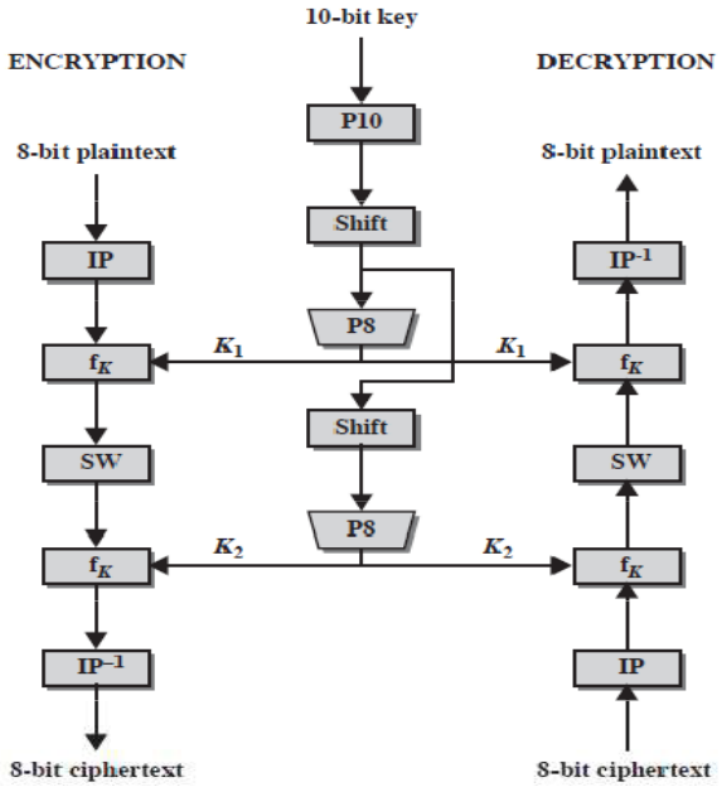

Figure 2 Simplified DES Scheme

You can find detailed example here: https://www.c-sharpcorner.com/article/s-des-or-simplified-data-encryption-standard/

Project Objectives:

In this design project, go through the different phases of design we discussed in the course. Start with system specification then design and verification. In your projects use a hierarchical structure of subsystems, so that subsystems can be initially developed and tested in incremental approach. This approach ensures that you will have working subsystems then work on the integration. During working on the design, make sure you take into account and apply the design methodologies we discussed during the course.

The final report should discuss:

- Research and Background of the project
- the function of the system
- system-level architecture
- subsystem design
- schematic and cell designs
- meaningful simulation results
- proposed testing methodologies
- lessons learned, problems faced, and unexpected findings.

The project can be broken down into the following phases

1) Behavioural stage: good understanding of the system and what you will design and how you will test it.
    - You can prepare a high-level, behavioral specification for your design. You can use C/C++, Matlab, VHDL, Verilog or any other behavioral specification method that can be simulated. Simulate your design and make sure that it works.
    - Try to estimate the sizes of various components of your design (eg. number of gates or transistors).

2) Implement the schematics of your blocks, ensure each block is functional alone and them start connect them.
    - Show simulation results of individual blocks and integrated system
3) Completed the layout of all your blocks and started routing and placement.
    - The entire layout should be complete and verified. All verification tools must have been run (DRC and LVS).
    - Run Parasitic extraction, back annotate and re-simulate, assess the impact of parasitics on your design.
    - Calculate total Area, power and throughput.
    - Compare actual area to estimated area

Number of Students:

4-6 Students, all students should understand the overall idea of the project and share in making the different design decisions.
Breaking down the block schematic and layout design can be done according to the number and agreement. Different responsibility of each student should be clarified in the final report.

Assessment Criteria:

The quality of the report is an important part of the project grade, consider it like writing a paper for submission in conference or journal to show your work.

- **The grade will be divided as follows:**
- **40% Proper functionality of block level and schematic**
- **20% Layout and physical verification**
- **30% Report and presentation Quality**
- **10% Creativity and novelty**

Best of Luck!