

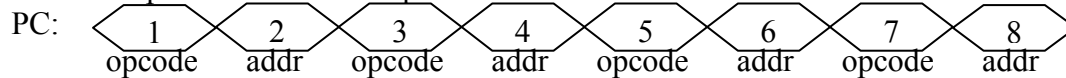
## ***Project Description***

### ***Mini-Computer Design(55)***

A 4-bit mini-computer with 8-bit instructions will be implemented. First four bits of the instruction determine the opcode, while the last four bits determine an address from data memory or a constant value according to the opcode.

In the system, there will be two different RAMs<sup>1</sup> which are used as Instruction Memory (IM) and Data Memory (DM).

A program counter (PC) will be used in the system. PC points to the next instruction in the IM and it is incremented in each clock cycle. Each instruction is read in two clock cycles. In the first cycle, PC points to the opcode, while in the second cycle, it points to the memory address or a constant value according to the opcode. Maximum value of PC is 2<sup>7</sup>. Sample of relationship between PC and opcode/address is as follows:



External Clock will be used. Its circuit will be given later.

There are four 4-bit registers (A,B,C and D), one 4-bit input port (SW4-SW1), and one 4-bit display (LED4-LED1) in the system. Instructions used in the system are as follows:

Opcode	Instr. Name	Address    Constant	Description
0000	LDAM	< addr >	Load the value in the memory to the register A
0001	LDBM	< addr >	Load the value in the memory to the register B
0010	LDAP	< 0 >	Load the value at the input port <sup>2</sup> to the register A
0011	LDBP	< 0 >	Load the value at the input port to the register B
0100	STC	< addr >	Store the value of Register C into the memory
0101	STD	< addr >	Store the value of Register D into the memory
0110	SHRCD	< 0 >	Circular right shift of the C:D <sup>3</sup> registers
0111	SHLCD	< 0 >	Circular left shift of the C:D registers
1000	ADD	< 0 >	C:D <-- A+B
1001	SUB	< 0 >	C:D <-- A-B
1010	AND	< 0 >	D <- A AND B
1011	OR	< 0 >	D <- A OR B
1100	DISP <sup>4</sup>	< Register ID >	Display the value of given register
1101	-----	-----	Open for your design
1110	-----	-----	Open for your design
1111	-----	-----	Open for your design

<sup>1</sup> Entity of RAM in Xilinx is RAMB4\_S4, sample code is added to the course web site.

<sup>2</sup>SW4-SW1 of the FPGA board.

<sup>3</sup>C:D represents the 8-bit register which is obtained from the concatenation of C and D registers

<sup>4</sup>LED4-LED1 of the FPGA board

CMPE 240  
Project  
Due: 11/06/2007 07:00 AM  
Demo: 11-12/06/2007

In the project, you will make a design for such a mini-computer, implement your design in vhdl and load your vhdl code into the FPGA.

**QUESTION-1 (5):** Give the instruction sequence to take the average of 2 numbers read from the input port, and display the result.

**QUESTION-2 (20):** Give the instruction sequence to apply linear interpolation to the numbers into the DM. Linear interpolation means that 1<sup>st</sup>, 3<sup>rd</sup>, 5<sup>th</sup>, 7<sup>th</sup> numbers are read from the memory, 2<sup>nd</sup>, 4<sup>th</sup>, and 6<sup>th</sup> numbers are calculated and written into the DM. 15 pt for operation, 5 pt for another instruction sequence to install the initial data (i.e. 1<sup>st</sup>, 3<sup>rd</sup>, 5<sup>th</sup>, 7<sup>th</sup>) into the DM.

**QUESTION-3 (20):** Give the instruction sequence to make a search in a list with 8 numbers into the DM for a number read from input port and display 1111 if it finds, 0000 if it doesn't. Matching control is done by using logic operations. 15 pt for operation, 5 pt for another instruction sequence to install the initial data into the DM.

**BONUS(10):** Give the instruction sequence to make multiplication.

You will submit all of your vhdl code and testbench waveforms with a report. Report will basically include:

- Assumptions
- Diagram and detailed explanation of the design
- Diagram and detailed explanation of each part of the system
- Answers for questions

Note: You will have only 5 minutes for demo. So, debugging will not be allowed during demo. It is strongly advised to test your code before demo section in the lab.

In the project, you should generate your clock signal by using following circuit. Input P stands for "Pulse" and R stands for "Reset". You will give these inputs manually by using buttons (BTN) on the FPGA board.

