CME 2206 – LAB PROJECT
2017-2018 SPRING

DESCRIPTION

You will design a basic computer DEUSEM (DEU Small Experimental Machine) which is similar to Mano's Basic Computer. DEUSEM has registers, memory, arithmetic-logic unit, control unit and bus system.

Quartus II software will be used to design and verify DEUSEM. The project is given as a term project and will be implemented in weekly lab sessions. It is advised that you read problem definitions of all of them before actually starting to implement your design, i.e., Common Bus and Registers.

Please submit zipped All Files (don't forget to submit waveform of the results) of the simulations for each lab session.

ASSIGNMENT 1 – COMMON BUS SYSTEM

REGISTERS

DEUSEM has 7 registers which are *Address Register, Data Register, Program Counter, Accumulator, Instruction Register, Input Register and Output Register.*

Register Symbol	*Register name*	*Number of bits*	*Function*
AR	Address Register	3	Holds address for data memory. It can be cleared with related control signal.
DR	Data Register	4	Holds memory operand
PC	Program Counter	4	Holds address of instruction. It can be cleared and increase by one with related control signals.
AC	Accumulator	4	Processor register
IR	Instruction Register	11	Holds instruction code
InpR	Input Register	4	Holds input data
OutR	Output Register	4	Holds output data

Table 1 - List of Registers for DEUSEM

MEMORY

In DEUSEM there is a memory that has three segments which are ***instruction*** and ***data segments***. Each has read enable and data inputs. Data memory segment has write enable input as an extra.

1. Instruction (Code) Memory Segment (16x11)
2. Data Memory Segment (8x4)

BUS SYSTEM

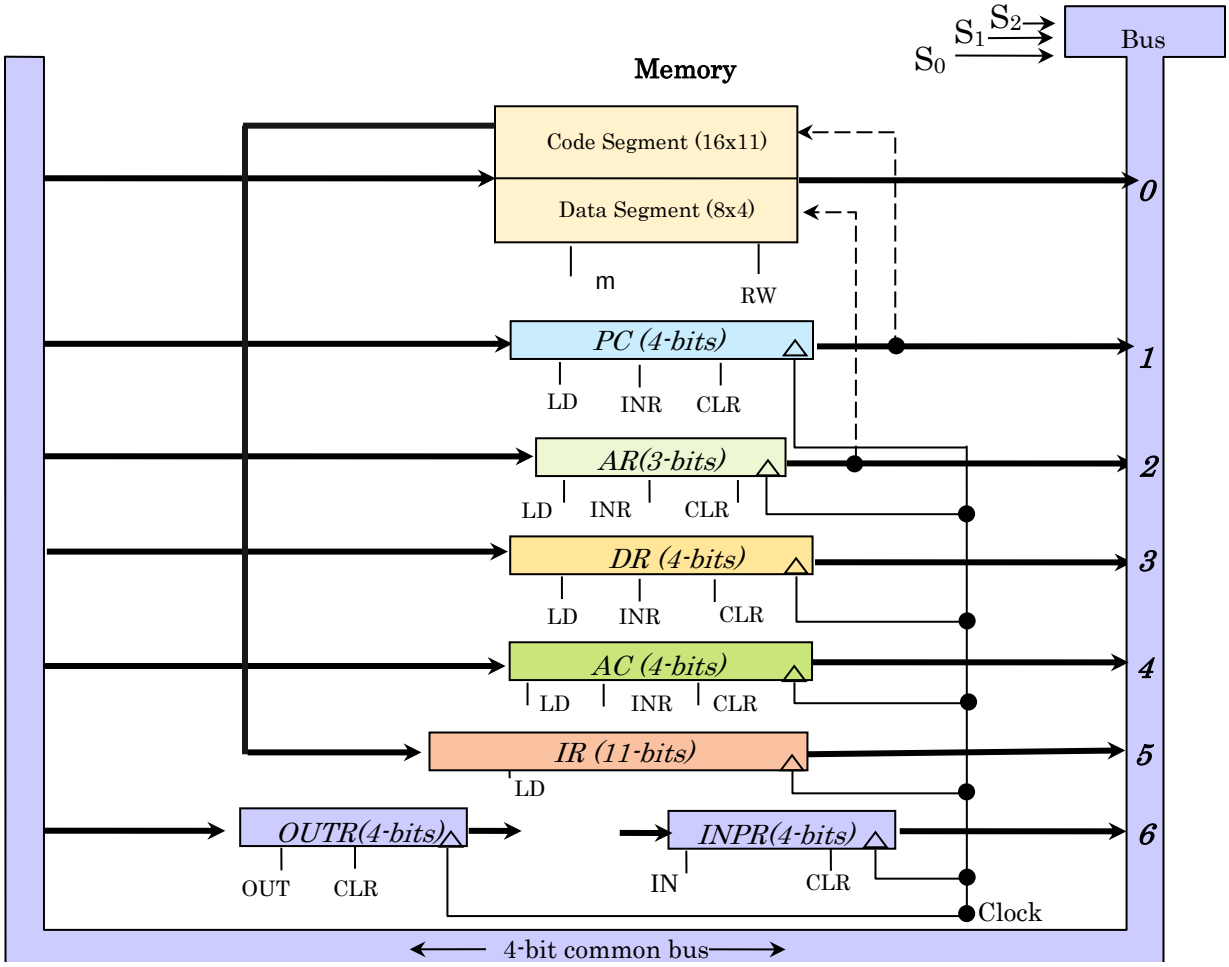

Figure 1 DEUSEM Common Bus System

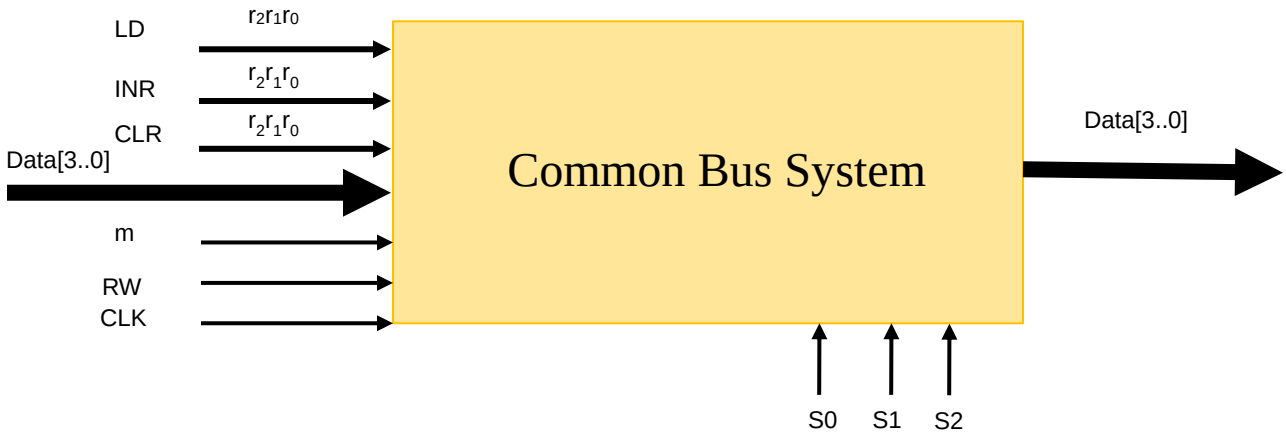

Figure 2 Block Diagram of DEUSEM Common Bus System

You are expected to implement the common bus architecture designed in Figure 1 in Quartus II and save it as a block diagram ('symbol file') with the name, "Common_Bus_System" as shown in Figure 2. Finally test and simulate your implementation by loading (transferring) data from Code memory and Data memory to registers and applying INR and CLR operations on them.

Details are listed below;

- There is one memory unit consisting of Code Segment and Data Segment Units. Selections are as follows

$S_2S_1S_0$	m	**Register / Memory Segment**
000	0	Code segment
	1	Data segment
001		PC
010		AR
011		DR
100		AC
101		IR
110		INPR OR OUTR

- Each bus control inputs of the bus block diagram (figure-2) indicated by $\mathbf{r_2r_1r_0}$ are encoded same as unit selections. That is,

`LD [`$r_2r_1r_0$ `= 1]` indicates PC's LD input

`LD [`$r_2r_1r_0$ `= 2]` indicates AR's LD input

....

`CLR [r2r1r0 = 1]` indicates PC's CLR input

`CLR [r2r1r0 = 2]` indicates AR's CLR input

....

`INR [r2r1r0 = 1]` indicates PC's INR input

`INR [r2r1r0 = 2]` indicates AR's INR input

....

`ReadWrite` and **`m`** applies only to memory unit

- Registers have 5 inputs which are data inputs, load enable input, clear input, clock input and increment enable input.
- Register sizes are given on the figure.
- Memory units must be initialized with preloaded memory files (`hex` or `mif`).