ELECTRICAL ENGINEERING

EEEE 220 – Digital Systems II

Lab6: Processor Design

Objective

The purpose of this laboratory exercise is to design the actual processor. That also includes Implementation and simulation. All of the components needed for the Datapath Design have been introduced in the previous Laboratory exercises **AND IN THE LECTURE!!!** The components needed are: (1) a register file which can be implemented using registers and logic for selection, (2) the ALU designed earlier and (3) memory blocks. You will also have to design an ASM in VHDL

Instruction Set Architecture

The Instruction Set Architecture, ISA, is enforced. The table below contains the (abstract) RTN description of each of the 16 instructions.

#	**4-Bit Instruction Code ***	**Instruction Mnemonic**	**RTN Description**	**Comments**
0	0000	ADD	A ← (A + B) : C-DFF ← C	Four-bit result gets stored in A. Carry out is stored in C-DFF.
1	0001	SUB	A ← (A – B) : C-DFF ← C	
2	0010	INC	A ← (A + 1) : C-DFF ← C	
3	0011	DEC	A ← (A – 1) : C-DFF ← C	
4	0100	NOT	A ← (/A)	/A := The complement of A
5	0101	AND	A ← (A • B)	
6	0110	OR	A ← (A + B)	
7	0111	SHR	$A_3 \leftarrow A_3 : A_2 \leftarrow A_3 : A_1 \leftarrow A_2 :$ $A_0 \leftarrow A_1 : C \leftarrow A_0$	Shift-Right; the sign bit is preserved; LSBit is loaded into the C-DFF.
10	1010	SWAP	A ← B, B ← A	
11	1011	CPY	B ← A	REG-A remains unchanged
12	1100	WR	M[[PC] ← A	Write contents of REG-A to memory location **
13	1101	RD	A ← M[PC]	Read contents of memory location into REG-A **

Created by Dorin Patru with the
assistance of Konboye Oyake
Upgraded by Yannis Papaefstathiou

Legend:
A<3..0> := A Register (4-Bits)
B<3..0> := B Register (4-Bits)
PC<7..0> := Program Counter (8-Bit) C-DFF := Carry D-FlipFlop – a single-bit register that holds the carry of the most recent arithmetic operation, or the LSbit of A during a SHR operation
IR<3..0> := Instruction Register (4-bit) Holds the opcode of the currently executing instruction.
M[0..255]<3..0> := Memory (256 4-bits words).

Table 5.1 – Instruction Set Architecture

(*) – The 4-bits instruction code is strictly enforced.
(**) – The memory address is computed as described further in the section “Memory Address Arithmetic”
(***) – These instructions are only implemented for processors with Memory Mapped I/O Peripherals. These processors must implement a 4-Bits/4-Location Stack.

Algorithmic State Machine (ASM) Chart for Processor

The control unit will generate the sequencing of the processor. Start with a preliminary ASM chart in order to determine the states and inputs to the Datapath which will be used to manually test the Datapath for correct functionality. Below is an excerpt of an ASM chart showing the machine cycles of one instruction, in particular the ADD instruction.

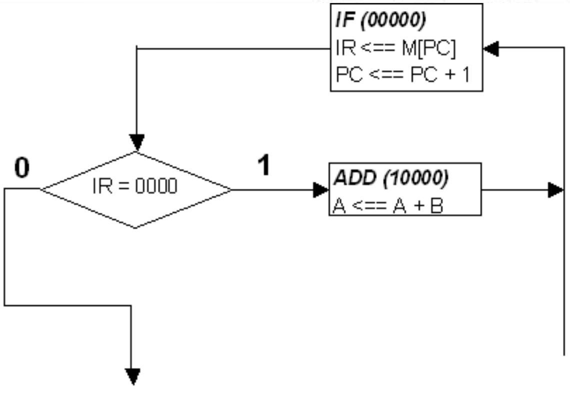
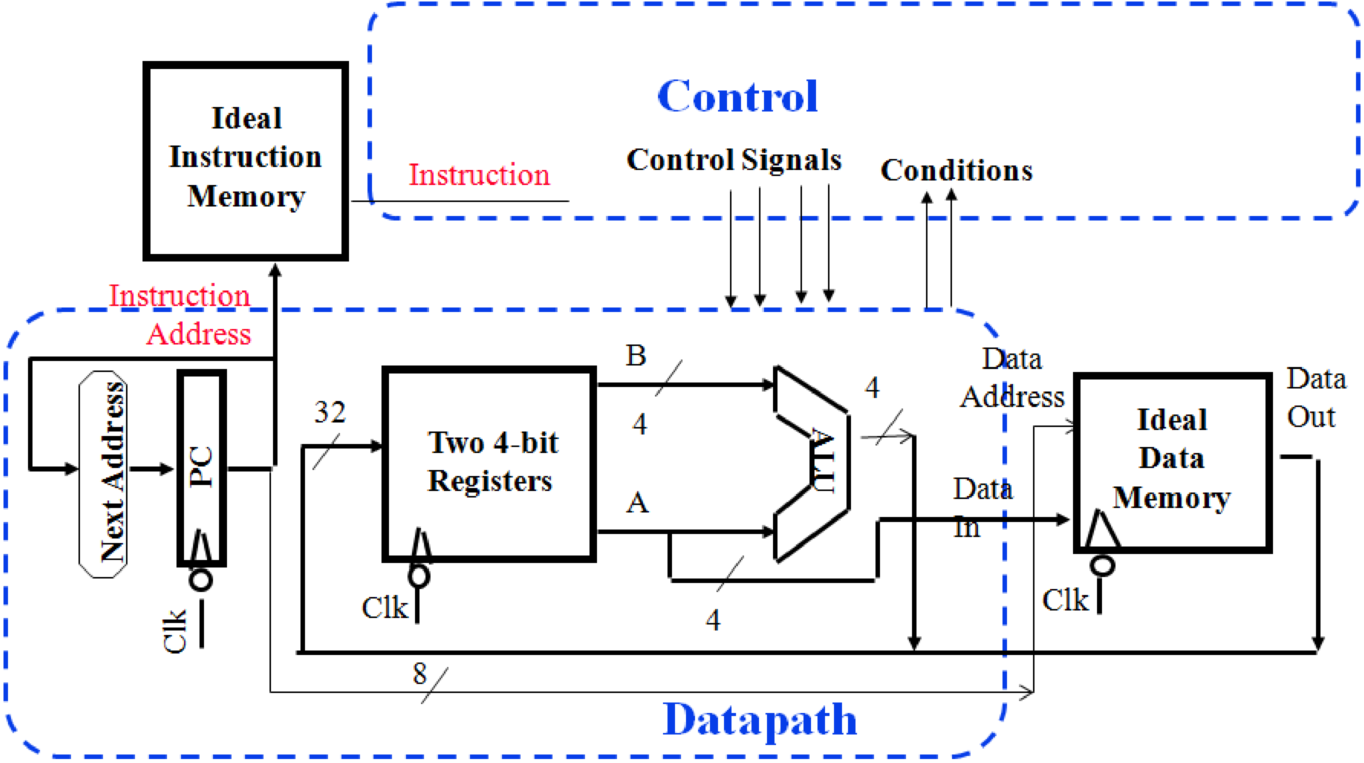


Figure 5.1 – ASM Chart for an instruction with a FETCH and one execution cycle.

Note: The numbers in the brackets, e.g. (00001) represent the state codes.

Complete the ASM Chart for all the 14 instructions specified and transalet it in VHDL and simulate it.



General Project Specifications

Below you can find the datapath and the control (which will comprise of your ASM) for the processor you will design

Register File

- Clock Synchronous for writing;
- Two 4-Bit Registers A and B, both with parallel load capability;

Function Unit - ALU

- ALU from Lab 5. Change the XOR function to accommodate the SHR-A operation.

Operation Select - FS[3..0]						
FS3	**FS2**	**FS1**	**CIN**		**Operation**	**Function**
0	0	0	0		F = A	Transfer A
0	0	0	1		F = A + 1	Increment A
0	0	1	0		F = A + B	Addition
0	0	1	1		F = A + B + 1	
0	1	0	0		F = A + (not B)	
0	1	0	1		F = A + (not B) + 1	Subtraction
0	1	1	0		F = A – 1	Decrement A
0	1	1	1		F = A	Transfer A
1	0	0	X		F = not A	NOT
1	0	1	X		F = A AND B	AND
1	1	0	X		F = A OR B	OR
1	1	1	X		F = SHR-A	SHR

Memory

- 8-Bit Addressable Memory, i.e. 8-bit wide Address Bus;
- 4-Bit Data-In-Bus;
- 4-Bit Data-Out-Bus;

- One **Program Memory** Address Space implemented **using an initialized ROM**.
- One **Data Memory** Address Space implemented **using an uninitialized, asynchronous RAM**.
- Asynchronous Write signal, MW = 1 to write, 0 to **not** write, i.e. read;

Program Counter - PC

- One 8-Bit Up Counter with parallel load capability;
- Clock Enable Signal;
- RESET Signal;
- Provides the address value to the instruction memory (segment), that points to the next instruction to be fetched and executed.

Laboratory /Project Requirement

You have to design, simulate, and emulate a central processing unit (CPU) that implements the instruction set described above in table 5.1. Follow the following steps:

1. Identify and write down your complete processor specifications;
2. Using two sheets of blank paper, start adding on one the building blocks you need to implement the 8 ALU instructions, while capturing on the second one the sequence of operations using an ASM chart. Continue with the other instructions.
3. After thoroughly scrutinizing it, present your design to the TA. The design must abide by all specifications assigned for your particular processor. The laboratory is not complete until the TA signs off on the design.

Your notes here: