Dalhousie University
Department of Electrical and Computer Engineering

ECED 3403 – Computer Architecture

Assignment 3: Implementation of the XM23p Instructions to access Data Memory

Objectives

XM23p is a load-store machine. In this assignment, we will design, implement, and test the XM23p emulator to support data memory (DMEM) and XM23's four data memory access instructions: LD and LDR (load from a memory location into a register) and ST and STR (store a register in a memory location).

Memory access using XM23 takes five cycles between instructions: two for the Fetch, one for the Decode, and two for the Execute:

Clock	Instruction	Stages	Comments
N+0	LD R1,R2	F0	Write PC to IMAR; Increment PC
N+1		F1	Read instruction from memory location
N+2		D0	Inst ← LD; SRC ← R1; DST ← R2
N+3		E0	DMAR ← R1; DCRTL ← RD-W
N+4		E1	R2 ← DMBR

In XM23p's pipeline implementation, an additional step is required in the Execute stage; this step allows the memory access to complete and either the memory to be written (the store instruction) or the memory to be read (the load instruction):

Clock	Instruction	Stages			Comments
		Fetch	Decode	Execute	
N+0	LD R1,R2	F0	D0		
N+1		F1		E0	Previous instruction is completed.
N+2	MOV R2,R0	F0	D0		
N+3		F1		E0	DMAR ← R1; DCRTL ← RD-W
N+4	ADD R1,R0	F0	D0	E1	R2 ← DMBR
N+5		F1		E0	R0 ← R2

XM23p takes three clock cycles from the end of one instruction to the end of a memory access instruction. This compares favourably with XM23's five clock cycles.

For more details on the XM23p design, please see the *XM-23p – The XM-23 Pipeline Processor Design Document* available in several locations on the course website.

Requirements

In this assignment, you are to implement a three-stage pipeline for XM23p as described in the *XM-23p – The XM-23 Pipeline Processor Design Document* for XM23p's four memory access instructions:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Mnemonic	Description
0	1	0	1	1	0	PRPO	DEC	INC	W/B	S	S	S	D	D	D	LD	DST ← DMEM [SRC plus addressing]
0	1	0	1	1	1	PRPO	DEC	INC	W/B	S	S	S	D	D	D	ST	DMEM [DST plus addressing] ← SRC
1	0	OFF	OFF	OFF	OFF	OFF	OFF	OFF	W/B	S	S	S	D	D	D	LDR	DST ← DMEM [SRC + sign-extended 7-bit offset]
1	1	OFF	OFF	OFF	OFF	OFF	OFF	OFF	W/B	S	S	S	D	D	D	STR	DMEM [DST + sign-extended 7-bit offset] ← SRC

The new instructions require changes to the Execute stage. See the XM23's ISA document for details on the four instructions.

Execute stage changes

The execution stage for the memory access instructions has two steps: E0 and E1.

Step E0

In step E0 during a memory access, the effective address is determined from the instruction:

Direct: The effective memory address is either the value of the source register for the load instruction (LD) or the value of the destination register for the store instruction (ST).

Indexed: The effective address depends on both the instruction (load or store), whether it is a pre- or post- increment or decrement, and whether the data is a word or a byte.

Pre- increments and decrements update the address register (the source register for loads or the destination register for stores), the updated register is then copied to the effective address. The size of the increment or decrement depends on the instruction's W/B bit; a word access changes the register by 2, while a byte access changes the register by 1.

Post-increments and decrements copy the address register (again, the source register for loads or the destination register for stores) to the effective address first. The address register is then updated, depending on the required action (increment or decrement) and whether it is a word (change the register by 2) or byte (change the register by 1) access.

Relative: The effective address is the sum of the source register (for loads, LDR) or the destination register (for stores, STR) and the seven-bit offset (sign extended, depending on the MSBit of the offset).

The effective address and the required registers are made available to step E1 through the DMAR, DCTRL, and DMBR.

Step E0 executes as it did in assignment 2, during the same clock tick as step F1.

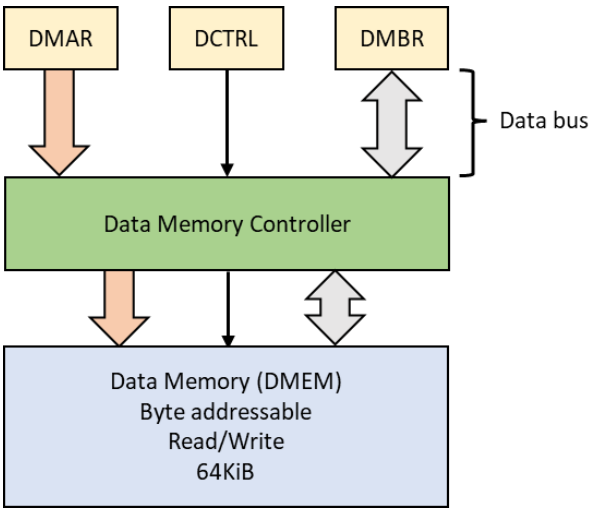
Step E1

Memory access instructions require a second step to allow the Data Memory Controller to complete the access (as with step F1 during the Fetch stage). At the end of E1, either the destination register (load) or the memory location (store) has been updated.

Step E1 occurs at the same time as steps F0 and D0. It is only executed if a data memory access is being performed.

Memory access

Your implementation will allow a program to access the machine's data memory, DMEM, with the addition of the four data memory access instructions LD, ST, LDR, and STR:



The DMEM will be accessed through the DMAR (Data Memory Address Register), DCTRL (Data Control register), and the DMBR (Data Memory Buffer Register). DMEM is byte-addressable, whereas

The DMAR holds a 16-bit address. The address can refer to an even- or odd-byte address.

The DCTRL is to indicate whether a read to occur (LD or LDR) or a write (ST or STR). It is to also indicate whether the access is for a word (two consecutive bytes) or a single byte; this is specified in the instruction's W/B bit. If a word access is taking place, the Data Memory Controller is to change the address in the DMAR to an even-byte address. A word-access returns a 16-bit word, while a byte-access returns an 8-bit byte from the location specified by the address.

Memory implementation

Memory can be implemented in at least two ways:

- As an overlay (union) of 32,768 words and 65,536 bytes. Bytes can be accessed using the DMAR value and the byte memory, while words can be accessed by shifting the DMAR value right by 1 (why?) and the word memory.
- As a single memory of 65,536 bytes. Bytes can be accessed using the DMAR value and the byte memory, while words must be built by concatenating an even-addressed byte as the LSByte and an odd-addressed byte as the MSByte.

Either approach is acceptable.

Data hazards

XM23p avoids most of the common pipeline data hazards. However, a data hazard can occur during a load when the destination register is the program counter. It will be necessary to design the pipeline to avoid this hazard. For more details on the XM23p's hazards, please see the *XM-23p – The XM-23 Pipeline Processor Design Document*.

Marking

The assignment will be marked using the following marking scheme:

Design: The design description must include a brief introduction as to the problem being solved (0.5), a design section (3), and a data dictionary (1.5).

Total points: 5.

Software: A fully commented, indented, magic-numberless, tidy piece of software that meets the requirements described above and follows the design description.

Total points: 11.

Testing: In addition to any test software supplied as part of the assignment, you will be responsible for developing sufficient distinct tests demonstrating that the software operates according to the design specification. Some of the tests should exercise the software. The tests must include the name of the test, its purpose or objective, the test configuration, and the test results.

Total points: 4.

All three sections are to be submitted electronically.

The loader must be demonstrated before the software and testing will be marked.

Important Dates

Available: 21 June 2024

Design document due: 2 July 2024 at 11:59pm, Atlantic.

Software and testing due: 9 July 2024 at 11:59pm, Atlantic.

Demonstration: 9 and 10 July 2024. Your implementation must be demonstrated before the software and testing will be marked.

Late submissions will be penalized 1 point every 24 hours.

Miscellaneous

This assignment is to be completed individually.

The work is to be implemented in C. It is recommended that you use Visual Studio for this assignment.

If you are having *any* difficulty with this assignment or the course, *please* contact Dr. Hughes or Emad as soon as possible.

This assignment is worth 10% of your overall course grade.

This is a non-trivial assignment. It should be started as soon as it is made available.

Assignments that are found to be copies of work done by other students or has used AI will result in an immediate dismissal from this course.