Assignment 1
The XM23p Loader and Memories
Design Document

Prepared for: Dr. Larry Hughes

Abdulla Sadoun B00900541

Table of Contents

Problem Introduction

Statement of Purpose

The purpose of this assignment is to design, implement and test a program written in C that would act as a standalone loader that shall be used to test the XM23p emulator developed by XMC.

Objectives

The program's primary objective is to take the s-records that have been obtained from the assembler as input. It would then read and interpret these records from the assemblers output (The .XME file containing s-records) It should be notes that the architecture used consists of 64KiB of code memory that is held separate from the other 64KiB which is dedicated for the data memory and parts of a debugger.

S-Records Data Dictionary

s-record = ‘s’ + type + length of record + Address + Data
Type = [0|1|2|9]
Length of record = Byte Pair
Address = 2[Byte Pair]2
Address = 0000-ffff
Data = 1[Byte Pair]30
Byte Pair = character + character
Character = 0-F

Pseudo Code:

Header file:

```
Remove the crt warnings
Include stdio
Include string

DEFINE DATASTARTINDEX 8

Function Prototype: ProcessSRec(line)
Function Prototype: Send2IMEM(address, byte)
Function Prototype: Send2DMEM(address, byte)
```

Implementation:

```
Include the header file

Function ProcessSRec(line){
    Declare type, count, address
    Initialize checksum=0

    // check the record is valid
```

```
If first character is not ‘s’ AND first character is empty AND record greater than 30
    print “invalid record”
    Return
End If

type = second character in record

If type = 0 then
    //process header record by decoding data to ascii name
    Declare name (empty string)
    For i from DATASTARTINDEX to end of line or not equal to null
        //Convert byte to ascii and store in array “name”
        Byte = hex to int algo
        Char = int as a character
        Add char to name string
    END For
    Print “name of the header record: ”, name

Else If type = 1 then
    RecordLength = convert hex to integers positions 2 to 4
    address = first two byte pair (positions 4 to 8) =hex to int
    Int Endindex = 8+(RecordLength-1 *2)

    For i from DataStart to EndIndex
        //load byte to IMEM
        Call Send2IMEM
        Increment address
        Checksum = checksum +byte
    End for loop
    If checksum not equal to -1
        Print “invalid record (checksum failure)”

Else if type = 2 then
    RecordLength = convert hex to integers positions 2 to 4
    address = first two byte pair (positions 4 to 8) =hex to int
    Int Endindex = 8+(RecordLength-1 *2)

    For i from DataStart to EndIndex
        //load byte to DMEM
        Call Send2DMEM
        Increment address
        Checksum = checksum +byte
    End for loop
```

```
                If checksum not equal to -1
                        Print “invalid record (checksum failure)

        Else if type =9 then
                Address = line position 4 - 8 ->hex to int algo.
                Print “execution address: ”, address
        Else
                Print “Invalid record type”
        End If

        return
End Function

Function Send2IMEM(address, byte)
        // Check address is within max limit
        If address <0 or >65536
                print “invalid IMEM address”
                return
        End If

        byteAddress = address *2  // since its a word address
        IMEM[byteAddress] = byte
        Print “loaded byte {byte} to IMEM address {address}”
End Function

Function Send2DMEM(address, byte)
        // Check address is within max limit
        If address <0 or >65536
                print “invalid DMEM address”
                return
        End If

        byteAddress = address *2 // since its a word address
        DMEM[byteAddress] = byte
        Print “loaded byte {byte} to IMEM address {address}”
End Function
```

Main:

Include header file

Main:
Take the xme file as input from CLI

```
Open .xme file for reading
    While (there are records)
        Read the content of the xme file line by line
        Call ProcessSRec(line)
    End while
    Close .xme file
End Main
```