Usage Documentation
Version 1.0

Dr. K. Muralikrishnan
kmurali@nitc.ac.in
NIT Calicut

August 29, 2014

Contents

Chapter 1

Introduction

This documentation describes the usage instructions for **SPL Compiler** (*System Programmers Language Compiler*), **APL Compiler** (*Application Programmers Language Compiler*) , **XFS Interface** (*eXperimental File System Interface*) and **XSM Simulator** (*eXperimental String Machine*).

Chapter 2

Setting Up

- Download the complete package http://xosnitc.github.com/fmyxos.tar.gz
- Copy the tar file to your home directory.

  ```
  cp myxos.tar.gz $HOME/
  ```

- Extract the contents using the command.

  ```
  tar -xvf myxos.tar.gz
  ```

  Now you will have a directory myxos in your home drectory, with all components required for building your own XOS

Chapter 3

SPL Compiler

3.1 Introduction

SPL (System Programmer's Language) Compiler is used in the implementation of an operating system on XSM (eXperimental String Machine). The compiler compiles the code written in SPL and translates it into machine code which is simulated on the machine.

3.2 Installation

3.2.1 Prerequisites

- GCC (GNU project C and C++ compiler)
- Flex / Lex (Fast Lexical Analyser Generator)
- Bison / Yacc (GNU Project Parser Generator)

3.2.2 Compiling and Running

Run the following commands to compile and run the SPL compiler

1. `make`
2. `./spl <flag> < <path-to-file>`

3.2.3 Flags

Any one of these flags is required to compile.

- `--os` : Compile OS Code
- `--int=timer` : Compile Timer Interrupt code
- `--int=[1-7]` : Compile Interrupt routines
- `--exhandler` : Compile Exception Handler

Chapter 4

APL Compiler

4.1 Introduction

APL (Application Programmer's Language) Compiler is used to write programs which can be run on XOS (eXperimental Operating System). The compiler compiles the program written in APL and translates it into machine code which is simulated on the machine.

4.2 Installation

4.2.1 Prerequisites

- GCC (GNU project C and C++ compiler)
- Flex / Lex (Fast Lexical Analyser Generator)
- Bison / Yacc (GNU Project Parser Generator)

4.2.2 Compiling and Running

Run the following commands to compile and run the APL compiler

1. `make`
2. `./apl < <path-to-file>`

The output file is named **"apcode.xsm"**

Chapter 5

XFS Interface

5.1 Introduction

XFS Interface (eXperimental File System) is an external interface to access the filesystem of the XOS. The filesystem is simulated on a binary file called "**disk.xfs**". The interface can format the disk, load/remove files, list files and copy blocks to a UNIX file.

5.2 Installation

5.2.1 Prerequisites

- GCC (GNU project C and C++ compiler)

5.2.2 Compiling and Running

Run the following commands to compile and run the interface

1. `make`
2. `./fileSystem`

5.3 Commands

Type the command `help` in the interface to display the list of commands.

5.3.1 Format the disk

The command **fdisk** is used to create the disk (”disk.xfs”) or to format the disk if already created. On a newly created disk or formatted disk the Disk Free List and FAT entries are initialized. Also a FAT entry is created for the file ”init.xsm”.
Syntax : `fdisk`

5.3.2 Load Files

The command **load** is used to load files to the filesystem from a UNIX file. The types of file that is loaded is specified by the first argument. The second argument <pathname> is the path to the UNIX file which is to be loaded to the filesystem.
The command checks the size of the file (executable or data files) and allocates blocks to it. The corresponding FAT entry is created for the file. For the OS Startup code, Interrupt routines and Exception Handler, the file is loaded to the corresponding location in the disk.

- *Syntax* : `load --exec` <pathname>
  Loads an executable file to XFS disk
- *Syntax* : `load --init` <pathname>
  Loads INIT code to XFS disk
- *Syntax* : `load --data` <pathname>
  Loads a data file to XFS disk
- *Syntax* : `load --os` <pathname>
  Loads OS startup code to XFS disk
- *Syntax* : `load --int=timer` <pathname>
  Loads Timer Interrupt routine to XFS disk
- *Syntax* : `load --int=[1-7]` <pathname>
  Loads the specified Interrupt routine to XFS disk
- *Syntax* : `load --exhandler` <pathname>
  Loads exception handler routine to XFS disk

5.3.3 Remove Files

The command **rm** is used to remove files from the filesystem. The first argument specifies the type of file to be removed. The argument <xfs_filename>

specifies the file which is to be removed.
The command searches the FAT entries for the file (executable/data file) and clears the blocks corresponding to the file. In the case of OS Startup code, Interrupt routines and Exception Handler the corresponding blocks in the disk are cleared.

- *Syntax* : `rm --exec <xfs_filename>`
  Removes an executable file from XFS disk

- *Syntax* : `rm --init <xfs_filename>`
  Removes INIT code from XFS disk

- *Syntax* : `rm --data <xfs_filename>`
  Removes a data file from XFS disk

- *Syntax* : `rm --os`
  Removes OS startup code from XFS disk

- *Syntax* : `rm --int=timer`
  Removes the Timer Interrupt routine from XFS disk

- *Syntax* : `rm --int=[1-7]`
  Removes the specified Interrupt routine from XFS disk

- *Syntax* : `rm --exhandler`
  Removes the exception handler routine from XFS disk

5.3.4 List Files

The command **ls** lists all the files which are loaded into the filesystem. The size of the file is also displayed in number of words.
The FAT entries are traversed and all the files and their corresponding sizes are displayed.
Syntax : `ls`

5.3.5 Display Disk Free List

The command **df** displays the disk free list. It also displays the total number of blocks and the number of free blocks.
Syntax : `df`

5.3.6 Display File contents

The command **cat** displays the contents of a file in the filesystem with the corresponding word number.
The FAT entries are searched to get the blocks corresponding to the file. The blocks are displayed.

Syntax : `cat <xfs_filename>`

5.3.7 Copy contents of File

The command **copy** copies the contents of specified blocks from a file in the filesystem to an external UNIX file. The arguments <start_block> and <end_block> denotes the range of blocks to be copied (including both). <unix_filename> specifes the destination UNIX file to which the contents are copied to.
Syntax : `copy <start_block> <end_block> <unix_filename>`

5.3.8 Display help

The command **help** displays the general syntax and function of all the commands.
Syntax : `help`

5.3.9 Exit Interface

The command **exit** quits the inteface.
Syntax : `exit`

Chapter 6

XSM Simulator

6.1 Introduction

The XSM (eXperimental String Machine) Simulator is used to simulate the XSM hardware

6.2 Installation

6.2.1 Prerequisites

- GCC (GNU project C and C++ compiler)
- Flex / Lex (Fast Lexical Analyser Generator)
- Bison / Yacc (GNU Project Parser Generator)

6.2.2 Compiling and Running

Run the following commands to compile and run the XSM Simulator

1. `make`
2. `./xsm <optional-flags> <path-to-file>`

The <optional-flags> can be :

- `--id` : This flag disables the timer interrupt for the machine
- `--db` : This flag sets the machine into DEBUG mode