Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science

6.111 - Introductory Digital Systems Laboratory

6.111 PROM Programming Tools – An Overview

This document describes an integrated set of tools which can be used to aid the programming of PROMs.

In digital design, PROMs are commonly used for the following purposes.

1) Control When used for control, the PROM is the program memory addressed by a sequencer. The output data is used to drive control lines and a sequencer is used to step through the machine instructions stored in the PROM memory.

2) Table-Look-Up Values of a function of a single variable can be stored in sequential locations in a PROM. The address corresponds to the input of the function and the data at that address corresponds to the output. Some calculators use such a table of logarithms so that they can do quick multiplication.

3) Characters The bit pattern for characters which need to be displayed on a CRT (cathode ray tube) can be stored in a PROM. As well as the regular characters, picture characters can be made for such things as space ships or other graphics often used in video games.

4) Logic Functions Logic functions can be implemented in a PROM as long as there are no more inputs to the function than address lines on the PROM.

The set of tools described in this document can aid a designer in programming a PROM to be used for one of the first three uses described above.

A short description of the programs in this tool-set and the files with which they work follows. A diagram which shows how they are connected is also shown. Detailed information is at the end of this document.

There is a set of shell scripts which can be used to run the programs. These scripts 'pipe' together (in the UNIX sense of the word) groups of programs which are usually run together. These scripts are described after the programs and files. A basic understanding of UNIX is assumed.

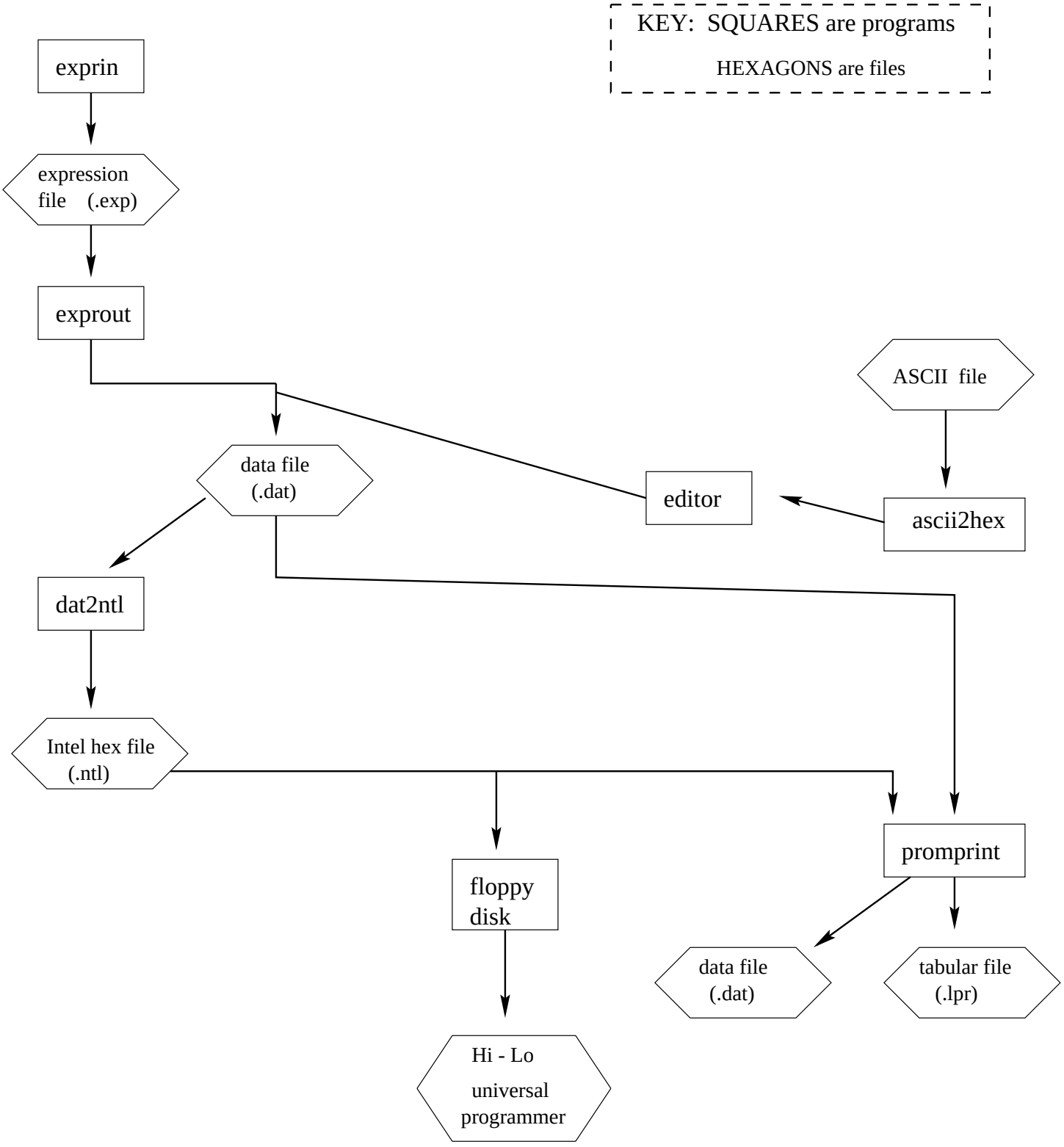

Figure 1: The PROM Programming Tool–set

PROGRAMS:

exprin >xxx.exp

An interactive program for creating an expression file.

exprout <xxx.exp >xxx.dat

Creates data from an expression file according to the expression, number of steps, starting address, etc.

dat2ntl [shift_count] xxx.dat xxx.ntl

Reads a file of integer data and command statements (e.g., #SET_ADDRESS = 0;) and creates a .ntl file in the format required by the PROM programmers. This program can also be used to shift and mask data.

promprint -[bhdcrn] xxx.ntl xxx.lpr

promprint -[bhdcrn] xxx.dat xxx.lpr

Produces a tabular version which is suitable for printing. The flag n takes the next argument to be the number of bits to print. The flags b, h, or d can be used to set the output base. The flags c and r can be used to change the number of columns and rows from the default which is eight. Each flag must be a separate argument. If input and output files are not specified then the standard input and output are used. The xxx.lpr format is equivalent to the xxx.dat format.

ascii2hex <infile >outfile

Translates ASCII characters into two digit HEX numbers. The resulting file is similar to a xxx.dat file except that it needs a #SET_ADDRESS command statement before being processed by the **dat2ntl** program. This must be done using a text editor.

FILES:

The (.letters) after the file name indicates the suffix which should be used with these files (e.g., logarithm.exp sin.exp). This naming convention is not required but, if used, makes it much easier to keep track of one's files.

expression file (.exp)
: Specifies information about a function and the steps the input should take. It can either be created by **exprin** or by using an editor – **exprin** is the easier way. The format that **exprout** requires for this file is given in the man page for **exprout**.

data file (.dat)
: A file of integers and command statements. The command statements include the SET_ADDRESS, RIGHT_SHIFT, MASK_COUNT, LOAD_ADDRESS, and BASE statements. You must include a #SET_ADDRESS or #LOAD_ADDRESS statement. This file, which may be created by 'assembler', is used by **dat2ntl** to create a file which is in the format required by the PROM programmer.

Intel hex file (.ntl)
: A formatted data file which is ready to be sent to the PROM programmer.

printing files (.lpr)
: The **promprint** program can be used to produce a tabular format file suitable for printing (e.g. with lpr). This (.lpr) file is equivalent to the (.dat) format.

SHELL SCRIPTS: See /mit/6.111/prom/scripts.

expression xxx

> Runs the interactive program **exprin** where you specify an expression and other relevant information. From this it creates xxx.ntl. It pipes together **exprin**, **exprout**, and **dat2ntl**.

expressdat xxx

> Runs the interactive program **exprin** where you specify an expression and other relevant information. From this it creates "xxx.dat". It pipes together **exprin** and **exprout**.

pview xxx.dat

> Displays a .dat file (or, indeed, any file), changing all 0's to blanks. This script is used to view characters or pictures which will be stored in a PROM.

Before programming a PROM, you must prepare an xxx.ntl file. This can be accomplished directly by assem or expression. Alternatively, you can prepare an xxx.dat file directly with an editor and use the dat2ntl program to produce the xxx.ntl file. After inserting your PROM into a socket, remember to turn down the lever on the socket to ensure good contact with the pins.

To program your PROM

For the time being, you should transfer your `.ntl` files to floppy and then use the universal programmers (the same ones used for PALs) to "burn" your PROMS.

SUMMARY OF PROM PROGRAMMING TOOLS

SHELL SCRIPTS
/mit/6.111/prom/documentation/latex/guide.1 PROGRAMS
/mit/6.111/prom/documentation/latex/guide2