GUIDE TO OPERATIONS AND TECHNICAL REFERENCE

Published by

LIMCO (PRODUCTS) MFG PTE LTD
109 KALLANG WAY 2
SINGAPORE 1334
REPUBLIC OF SINGAPORE

SECOND EDITION (MAY 1985)

PREFACE

This manual comprises of two parts, namely a GUIDE TO OPERATIONS and a TECHNICAL REFERENCE.

GUIDE TO OPERATIONS provides you with introduction to the LINGO PC–128 mkII microcomputer. Included are instructions for installations, using and programming your LINGO PC–128 mkII. This guide has 10 sections.

Section 1	**Introduction,** highlights the various components that make–up the LINGO PC–128 mkII computer system.
Section 2	**Setup,** contains step–by–step instructions on how to install your new LINGO PC–128 mkII computer system.
Section 3	**Disk and Disk Drives,** discusses correct disk and disk drives operations.
Section 4	**Keyboard,** describes the various features of LINGO PC–128 mkII keyboard.
Section 5	**Introduction to CP/M 3,** introduces to you the power of CP/M 3.
Section 6	**CP/M 3 Commands,** highlights the types of commands used under CP/M 3.
Section 7	**CP/M 3 Files,** describes the files naming conventions used.
Section 8	**Console and Printer,** shows how CP/M 3 control the console and printer.
Section 9	**Command Summary,** includes in–depth descriptions of all the CP/M 3 commands used.
Section 10	**DOS Version 3.3,** briefly summarises the use of DOS 3.3.

TECHNICAL REFERENCE is your guide to the "soul" of the LINGO PC–128 mkII Personal Computer, and is intended for those users who are keen on getting to know the inside of the computer. It has been written on the assumption that the user has some fundamental knowledge on computer hardware and software. This guide has 8 sections.

Section 11	**System Organisation,** introduces you the various functional blocks of the LINGO PC–128 mkII Computer system.
Section 12	**Memory Organisation,** describes the memory organisation of the system, namely the RAM and ROM, and the bank switching techniques.
Section 13	**Video,** covers the video outputs of the LINGO PC–128 mkII and the various display modes that are available.
Section 14	**I/O Organisation,** describes the input/output of the computer: the speaker, game I/O, peripheral slots, keyboard, etc.
Section 15	**Z80 Microprocessor,** explains how the Z–80 microprocessor is used as a co–processor to run CP/M programs.
Section 16	**Printer Interface,** describes how graphics dumpings and text printing can be acheived.
Section 17	**Serial Interface,** shows how the serial interface can be connected to a serial printer, and the different modes of operation.
Section 18	**Jumper Selection,** explains how the system board can be re–configured to cater for different modes of operation.

Appendix A	lists the Special I/O Locations of the computer, as well as an ASCII table.
Appendix B	shows how the motherboard can be re–configured for NTSC monitors.
Appendix C	contains the Z–80 microprocessor instruction set.
Appendix D	contains the 6502 microprocessor instruction set.
Appendix E	contains the ACIA 6850 specifications.
Appendix F	comprises of the system board layout and schematics.
Appendix G	lists the various optional peripherals and devices that are available for your system expansion.

We hope that this manual will assist you in getting the most out of your LINGO PC–128 mkII computer system.

CONTENTS

GUIDE TO OPERATIONS

Section 1. INTRODUCTION

Section 2. SETUP

Section 3. DISK AND DISK DRIVES

Section 4. KEYBOARD

Section 9. COMMAND SUMMARY

Section 10. DOS VERSION 3.3

TECHNICAL REFERENCE

Section 11. SYSTEM ORGANISATION

GUIDE TO OPERATIONS

SECTION 1 INTRODUCTION

Contents **Page**

INTRODUCTION

The LINGO PC–128 mkII is an 8–bit microcomputer system designed specifically to make learning, working, and playing time more of what it should be – rewarding, creative, and exciting. Whether you are a manager, engineer, computer enthusiast or even a first–time computer user, you will be impressed with the powerful capabilities and ease of operation. You will discover that the LINGO PC–128 mkII provides significant expansion options to suit your needs when you are ready to grow.

The LINGO PC–128 mkII is the successor of the ever popular LINGO PC–128. It provides these standard features:–

- Dual microprocessor (6502 and Z80A)
- 128K RAM
- Software selectable 80 column or 40 column text display
- Standard Centronics printer interface with graphics dump capability
- RS232C serial interface (CCS 7710D compatible)
- Multi–video outputs
  - Composite B/W
  - PAL or NTSC composite colour
  - RGB colour
  - Home Television UHF video
- Five I/O expansion slots
- Two types of game joystick connector
- Typewriter style, full ASCII keyboard
  - 90 keys
  - Upper and lower case
  - Numeric pad
  - Auto–repeat keys
  - Cursor control pad
  - Fixed and programmable function keys

SYSTEM CONFIGURATION

In this section, we will introduce to you the basic components that make up your LINGO PC–128 mkII microcomputer system. We will also highlight on the various operating systems that this computer support.

FIG. 1–1 A typical LINGO PC–128 mkII System

HARDWARE

'Hardware' refers to the physical parts of the computer system. The hardware of a typical LINGO PC–128 mkII computer system consists of the System Unit, dual disk drives, keyboard and a video display. Other optional devices that can be attached to the System Unit includes printer, game joystick, modem, and peripheral cards.

THE SYSTEM UNIT

The heart or brains of your computer system is the system unit. Under the cover of this powerful table–top unit lies the System Board which manages all activities and controls the flow of information between each and every devices that are attached to it.

The System Unit consists of the following parts.

(a) The System Board
(b) Dual disk drives with controller card
(c) Power Supply Unit

(A) THE SYSTEM BOARD

The System Board, which is commonly referred to as the 'Motherboard' is the main circuit board housing the integrated circuit chips and other electronic components.

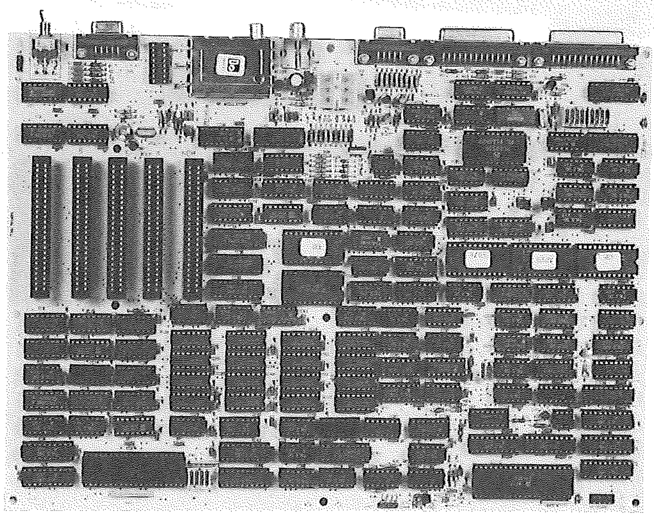


FIG. 1–2 System Board

The Motherboard can be categorised into the following functional parts:–

a) The microprocessor

The LINGO PC–128 mkII uses 2 microprocessors, namely the 6502 and Z80A. The 6502 microprocessor is incorporated to run APPLE programs while the Z80A supports CP/M operating system.

b) Read Only Memory (ROM)

A 2K monitor program performs the housekeeping of the computer. For example, it takes care of displaying whatever is typed on the keyboard to the video display. The monitor program is permanently stored in the ROM and is non–volatile, i.e., it is not erased even when the power is off.

c) Ramdom Access Memory (RAM) RAM)

The LINGO PC–128 mkII has a memory storage capacity of 128K which user or application programs could utilize. These memory areas are volatile, i.e., the contents will be erased when the power is off.

d) Parallel printer interface

A centronics parallel printer interface allows a parallel printer to be attached. It has the capabilities of printing both text and graphics.

e) Serial interface

Peripheral devices such as serial printer, modems and even other computer can be connected to this standard RS–232C serial interface.

f) 80 column

Your LINGO PC–128 mkII can display in both 40 characters and 80 characters per line. This gives you a software selectable 80 or 40 column text display. A switch labelled "80 COLUMN" can be used to switch off the 80 column mode.

g) RGB video

An RGB colour monitor can be connected to produce high resolution graphic display.

h) B+W/PAL

A typical monochrome monitor can be connected to display high resolution text and graphics. A PAL colour monitor can also be used.

j) RF modulator

An RF modulated PAL video is available for direct connection to your HOME TV.

k) GAME I/O interface

This allows game paddles or joystick to be attached to your System Unit.

l) Expansion Slots

There are 5 expansion slots on the system board marked as ROM, 4, 5, 6, 7 as shown. Slot 6 is presently occupied by a disk drive controller card. The ROM slot allows user–supplied BASIC ROM card to be inserted.

(B) DISK DRIVES

The primary function of a floppy disk drive is to store and retrieve information on a magnetic media, which is known as a floppy diskette. Programs and data in memory can be stored on the disk. A disk drive controller card, occupying slot 6, controls and tranfers data from the System Unit and the disk drives.

(C) POWER SUPPLY UNIT

The power supply unit provides all the necessary voltages to the system unit and the keyboard.

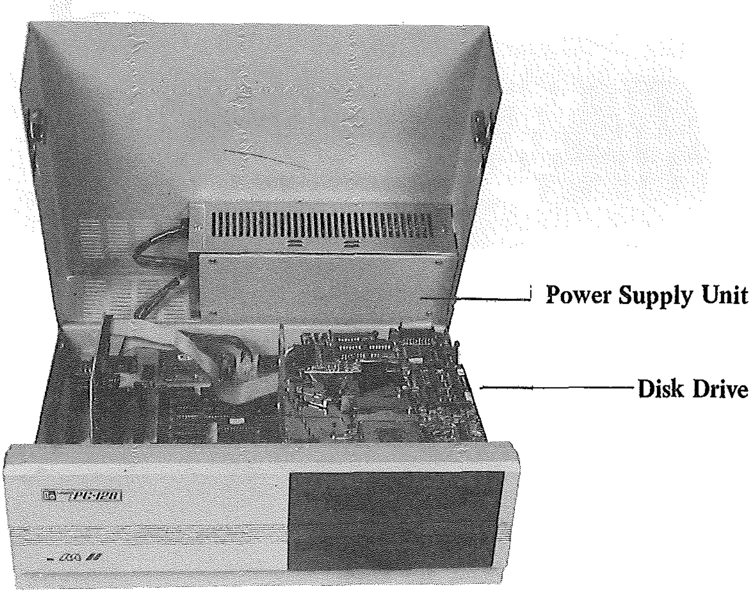


FIG. 1–3 Inside view of the System Unit

KEYBOARD

The keyboard is an entry device which allows you to communicate with your LINGO PC–128 mkII computer. It is a 90–key keyboard which features a numeric keypad and 10 function keys. The keyboard is connected to the System Unit by a one metre coiled cable, which allows you to move the keyboard freely to a comfortable operating position.

FIG. 1–4 Keyboard

VIDEO DISPLAY UNIT

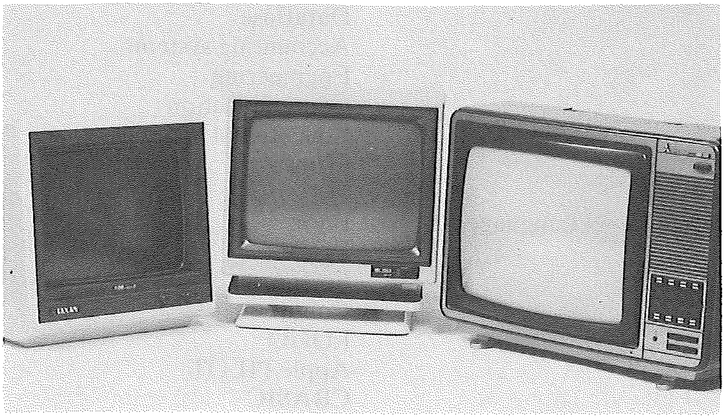
A video display unit (VDU) is an output device that the computer uses to display information. Any one of the followings can be used as a VDU for LINGO PC–128 mkII.

a) Monochrome (Green or Amber) monitor

b) PAL colour monitor

c) RGB colour monitor

d) Home Television Set



RGB Monitor **Monochrome Monitor** **Home TV**

FIG. 1–5 Video display units

SOFTWARE

With the combination of 6502 and Z80A microprocessors, a tremendous amount of 8–bit software is available for the LINGO PC–128 mkII. The software can be boldly classified as follows:–

a) Operating Systems	–	Apple Dos CP/M 2.2 CP/M 3.0 UCSD–P system Apple Pascal etc.
b) Application software	–	Word processing Electronic Spreadsheet DataBase Accounting systems Engineering Communication Educational Games
c) High Level Languages	–	Basic Pascal Fortran Cobol LOGO Apple PILOT CBASIC

TECHNICAL SPECIFICATIONS

1. CENTRAL PROCESSOR UNITS (CPU)
   - 6502 microprocessor running at 1 MHz.
   - Z80A microprocessor running at 2 MHz.

2. MEMORY SIZE
   - Ramdom Access Memory (RAM)
     128K dynamic RAM
     2K static RAM
   - Read Only Memory (ROM)
     2K Monitor
     2K 80–column display firmware
     4K Parallel printer firmware
     ¼K Serial I/O firmware
     2K Character generator codes

3. INPUTS AND OUTPUTS
   - Centronics parallel printer interface
     Advance text features allowing control of margins, page length and text screen dump.
     High resolution dot graphics dump routines allowing double size, inverse, and rotation of both HIRES pages.
     Supports DOS, PASCAL and CP/M Operating Systems.
   - RS-232C Serial I/O
     Compatible to CCS-7710D standard
     15 hardware selectable baud rates
     Terminal and Non-terminal firmware routines
   - 80 COLUMN DISPLAY
     Software selectable
     Emulate the Videx Videoterm 80 column card
   - I/O Expansion Slots
     4 Apple II compatible slots
     1 ROM slot
     Fully buffered with interrupt and DMA priority structure
   - Game Joystick/Paddles
     4 annunciator outputs
     4 analog inputs
     3 switch inputs
     D-9 female connection
     16-pin socket connection

4. VIDEO DISPLAY
   - Composite B/W
   - PAL or NTSC Composite colour
   - RGB colour
   - Home Television UHF Channel 36 video
   - Display modes
     - Text
       - 40 columns by 24 lines, 5x7 dot matrix
       - 80 columns by 24 lines
     - Low resolution colour graphics
       - 16 colours
       - 40x48 resolution or 40x40 with 4 lines of text
     - High resolution colour graphics
       - 6 colours: black, white, violet, green, blue and orange
       - 280x192 resolution or 280 x160 with 4 lines of text

5. KEYBOARD
   - Full ASCII
   - 90 keys
   - Upper and lower case
   - Numeric pad
   - Auto–repeat keys
   - Cursor control pad
   - 10 fixed and programmable function keys
   - Audible key touch

6. DISK DRIVE
   - Two built–in disk drives
   - 160K each
   - Silent operation
   - Direct driven spindle motor
   - 48 TPI Track Density
   - 40 Tracks

7. POWER SUPPLY UNIT
   - High efficiency switching type
   - 110/220V, 60/50Hz selectable
   - 80 watts capacity
     - +5V -- 5A
     - +12V -- 4A
     - −5V -- 0.5A
     - −12V -- 0.5A

8. DIMENSIONS
   - System Unit
     355mm x 325mm x 115mm
     10.5 kg (with 2 drives)
   - Keyboard
     450mm x 195mm x 35mm
     1.5 kg

SECTION 2 SETUP

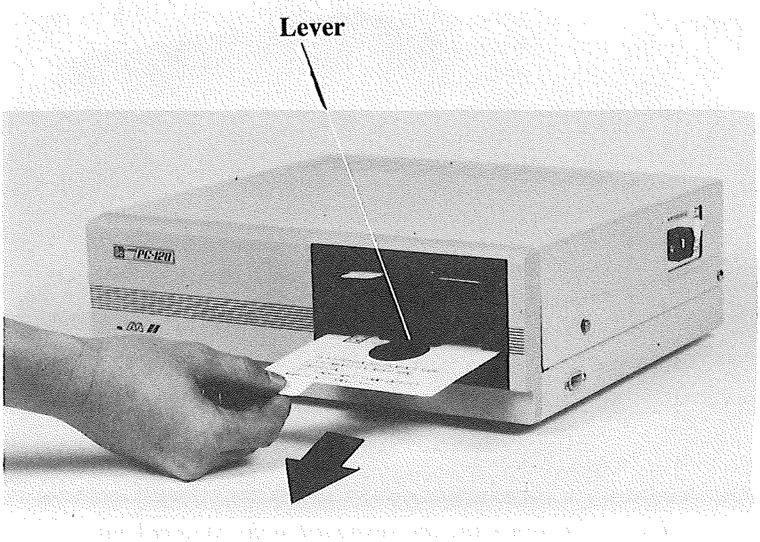
Contents **Page**

INSTALLING LINGO PC–128 mkII

1 REMOVING SHIPPING CARDBOARDS.

To open the drive door, press the lever down until a click sound is heard. Release the lever. The shipping cardboard, that protects the drive, will be ejected automatically. Remove the shipping cardboards from both drives.

SYSTEM UNIT



Remove Shipping Cardboard

FIG. 2–1 Removing shipping cardboard from the drive

2 Place keyboard in front of the System Unit and connect cable as shown.

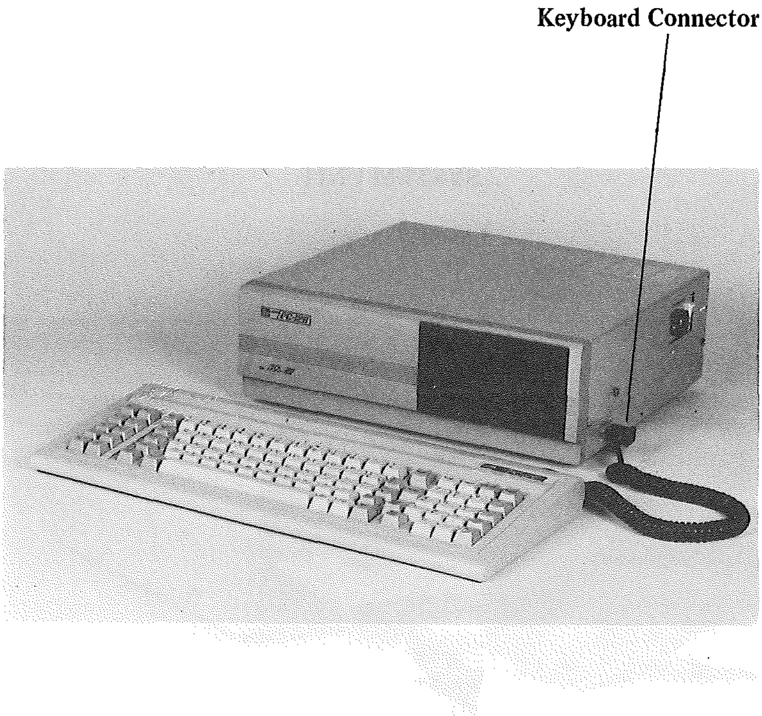


FIG. 1–2 Connecting the keyboard to the System Unit

3 Position System Unit power switch to OFF.

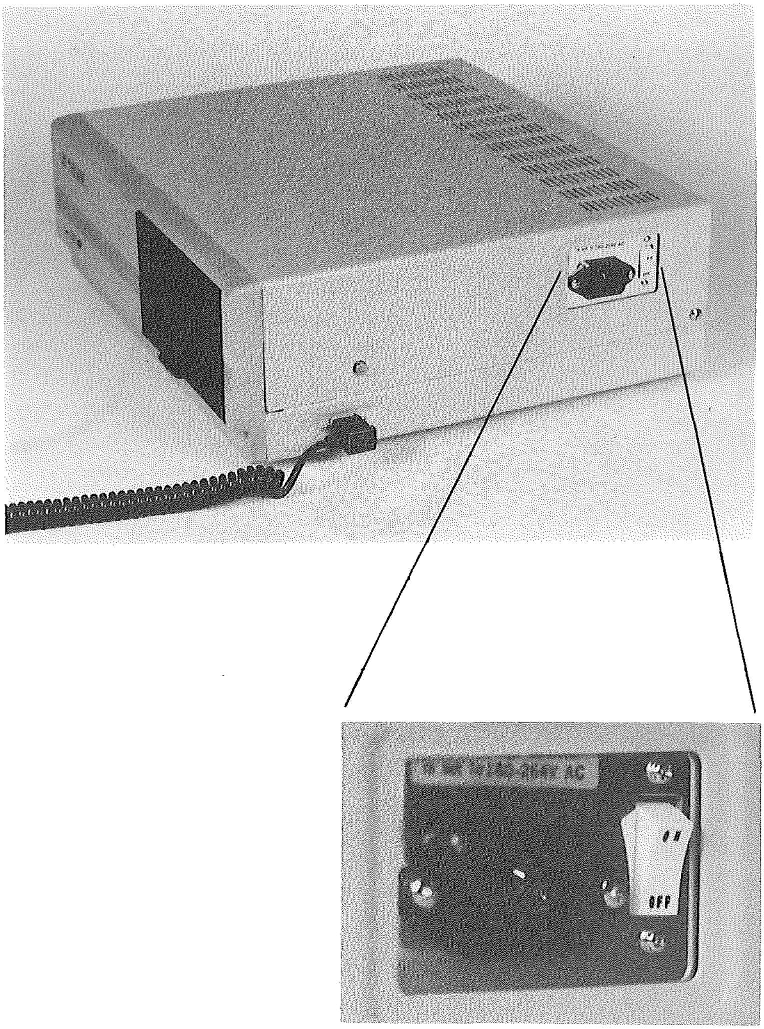


FIG. 2–3 Power switch in OFF position

4 Connect System Unit power cord to System Unit first, then to the wall outlet. Switch on the wall outlet power.

FIG. 2–4 Connecting the power cord to the System Unit

5 Preliminary Test

Position System Unit power switch to ON. There should be a short beep. At the same time, the power indicator light and the Drive A "IN USE" light will light up.

If your System Unit does not response correctly, consult your dealer.

Switch off your System Unit.

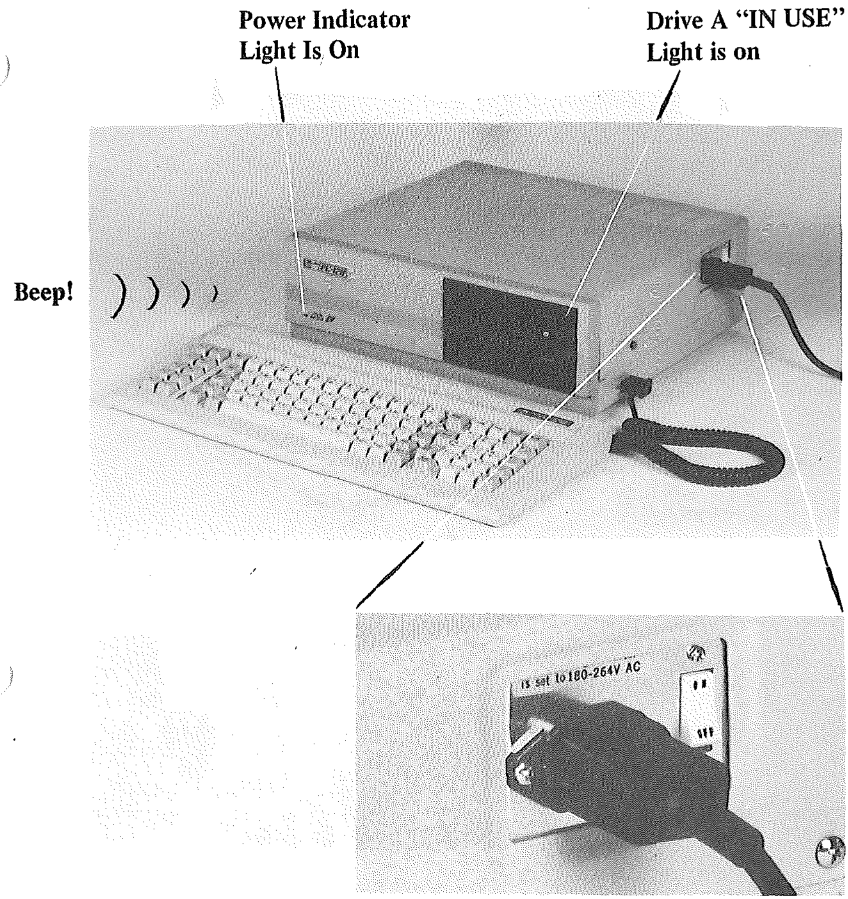


FIG. 2–5 The responses of the System Unit during the test

6 Installing Peripheral Cards

If you are not installing any peripheral cards, proceed to Step 7.

WARNING: Ensure main power is off before proceeding.

To open the System Unit cover, depress the two buttons located on both sides of the System Unit and lift it up gently.

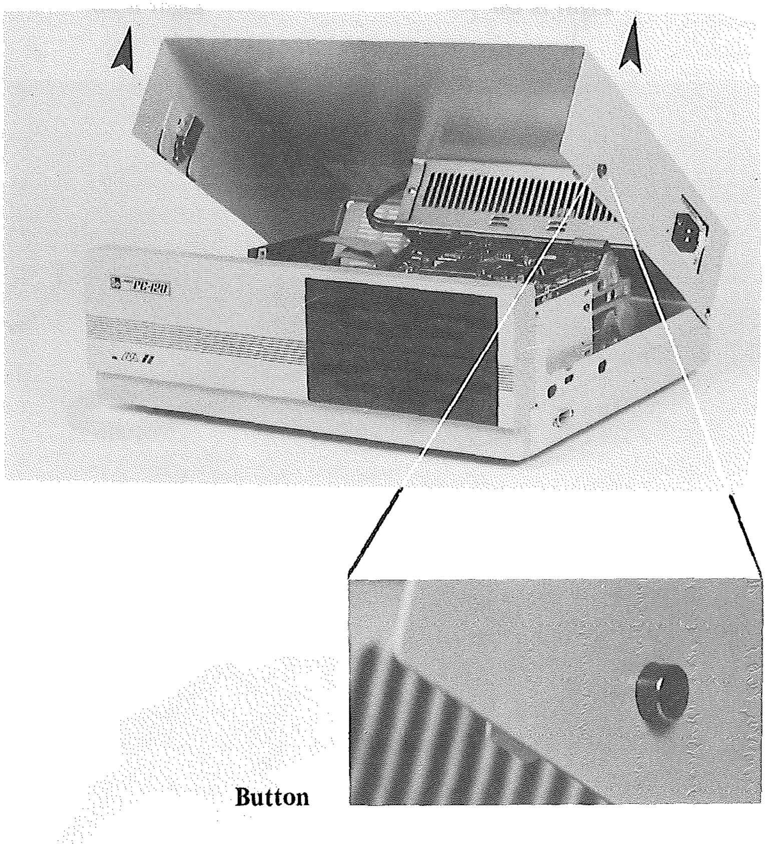


FIG. 2–6 Opening the System Unit cover

How to insert Basic Rom Card

Locate for the slot labelled "ROM".

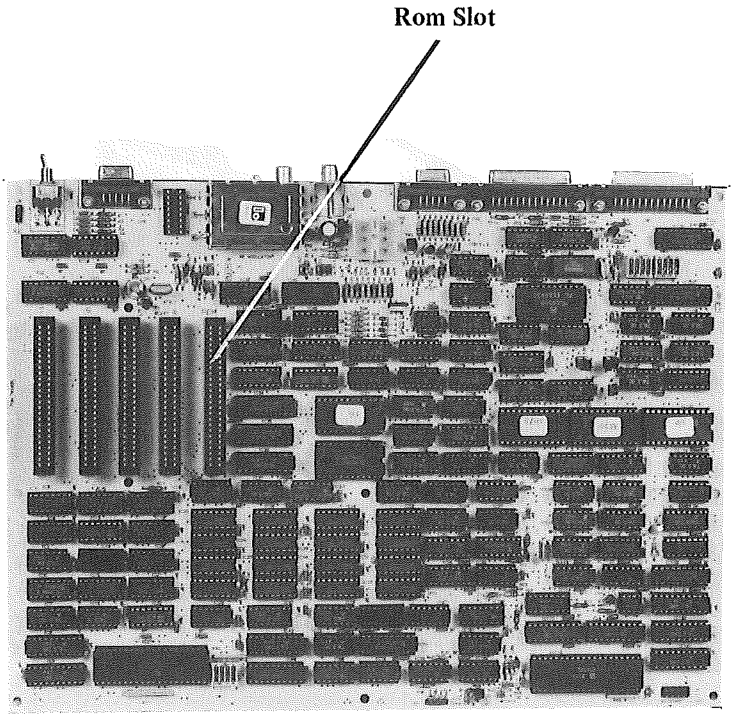


FIG. 2—7 ROM slot

Insert the card by a rocking movement until it is firmly seated. The card should be inserted with the components facing the drives as shown.

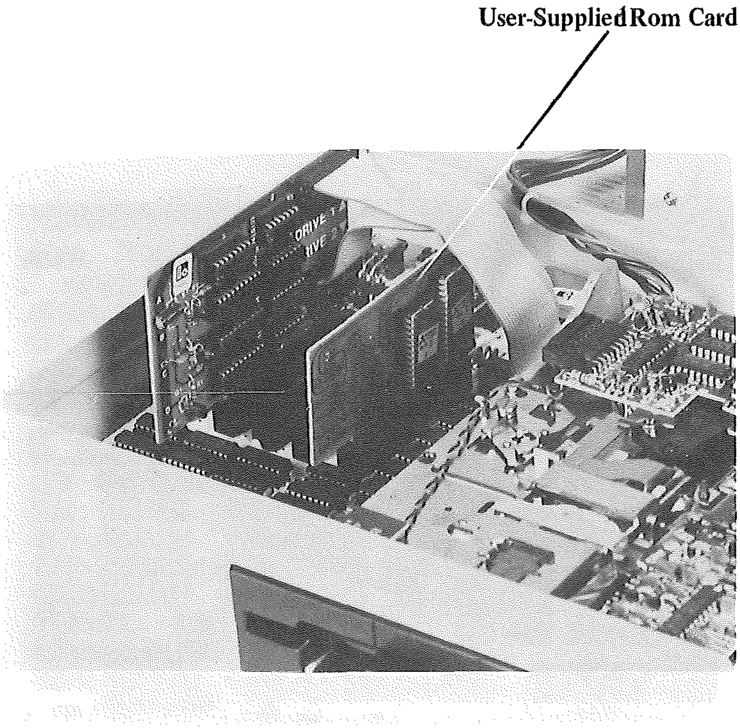


FIG 2–8 A BASIC ROM card seated in the slot

Other Peripheral Cards

To insert other peripheral cards, locate the required slot number (refer to your dealer for the correct slot number). Insert the card in a similar manner as described in the proceding section on How to insert Basic Rom Card.

Closing the System Unit cover

Lower the cover gently. Press the cover down on both sides until it is properly seated.

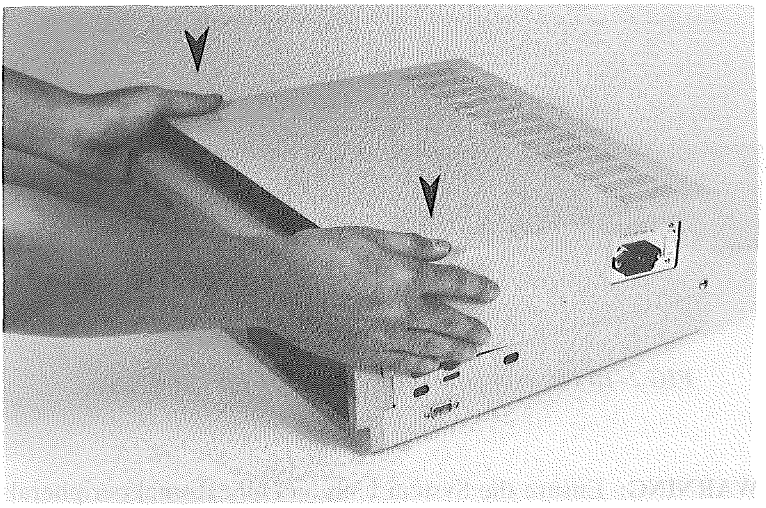


FIG. 2–9 Closing the cover

Rear Panel Referance

Use the handy reference diagram below as a guide to your installation of external peripherals.

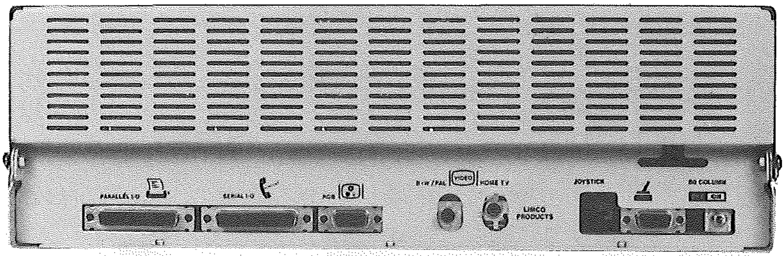


FIG 2–10 The rear panel of the System Unit

WARNING: Ensure the System Unit and all external peripherals are switched OFF before any installation is done.

The manufacturer shall not be responsible for any damages or/and injuries sustained in the cause of improper installation procedures.

7 Connecting the Video Display Unit

The LINGO PC–128 mkII can be connected to 3 main types of monitors; namely monochrome monitor, RGB monitor and HOME TV.

How to connect a Monochrome Monitor

Locate the video socket labelled "IN" of the monochrome monitor.

Locate the video output connector of the System Unit labelled "B+W/PAL".

Connect a video cable as shown.

Connect the monitor power cord to the wall outlet.

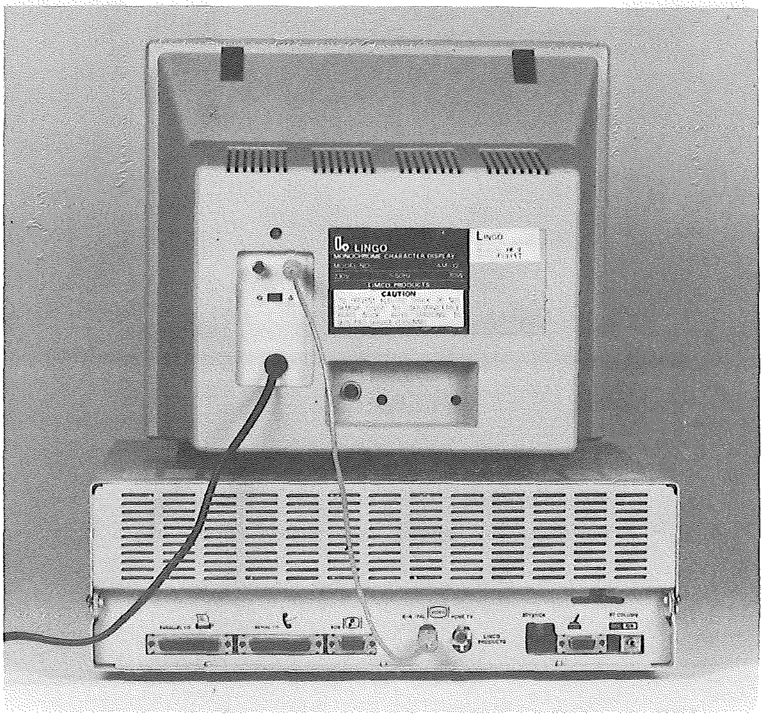


FIG 2–11 Connecting a monochrome monitor to the System Unit

How to connect a RGB Monitor

Locate the video socket of the RGB monitor.

Locate the video output connector of the System Unit labelled "RGB".

Connect a RGB video cable as shown.

Connect monitor power cord to the wall outlet.

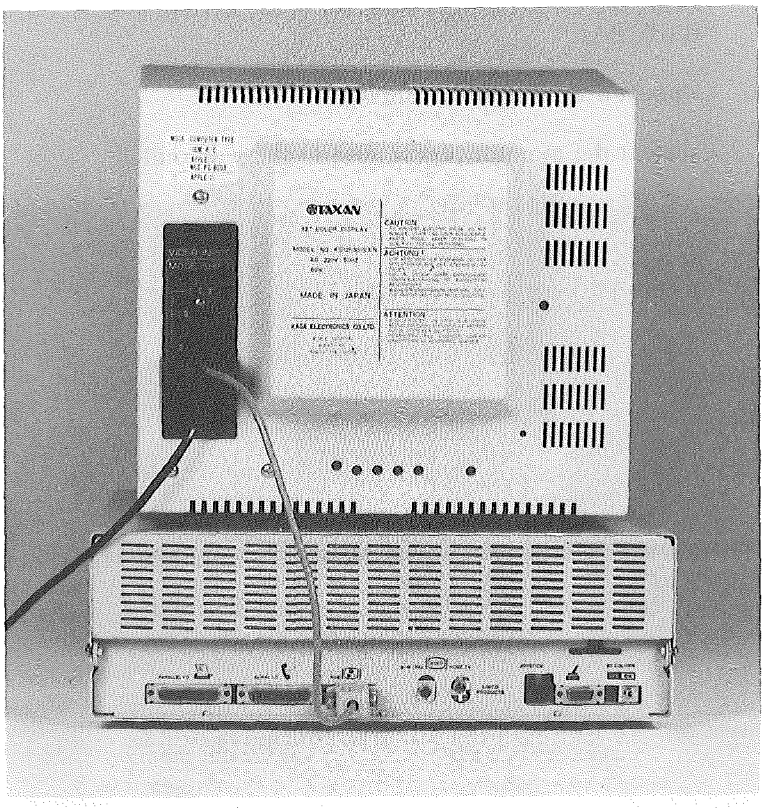


FIG. 2–12 Connecting a RGB monitor to the System Unit

How to connect a HOME TV

Locate the antenna input of the HOME TV.

Locate the video output connector of the System Unit labelled "HOME TV".

Connect a video cable as shown.

Connect the HOME TV power cord to the wall outlet.

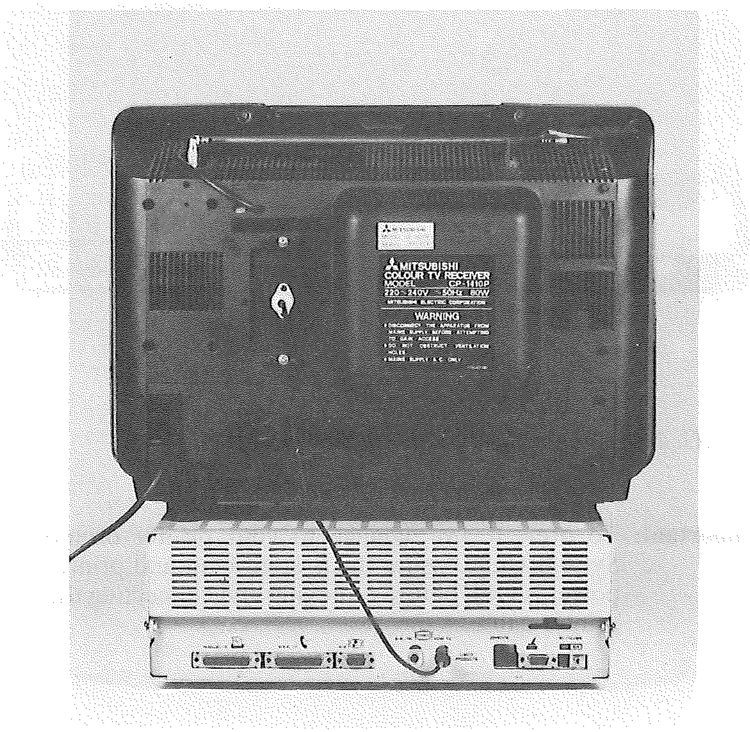


FIG. 2–13 Connecting a TV to the System Unit

8 How to connect a Centronics Parallel Printer

Locate the I/O connector of the printer.

Locate the I/O connector of the System Unit labelled "PARALLEL I/O".

Connect a printer cable as shown.

Connect printer power cord to the wall outlet.

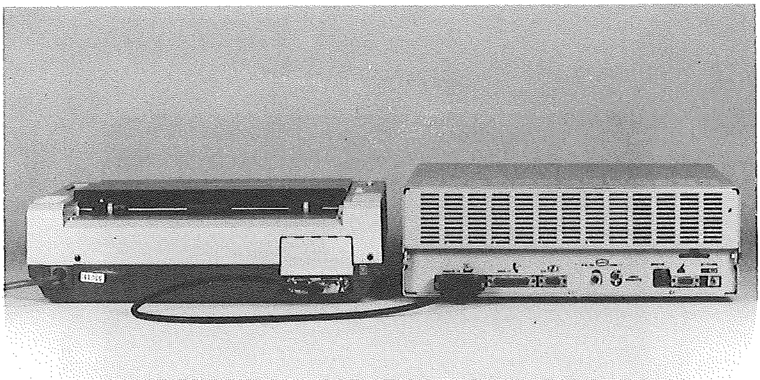


FIG. 2-14 Connecting a printer to the System Unit

Important: To prevent any damage to your printer or/and System Unit, ensure the System Unit and printer power is OFF before removing and connecting the printer cable.

9 How to connect a Joystick

There are two main types of joystick connection commonly used. Your LINGO PC−128 mkII has both types of connectors available. The two types of connectors are located as shown.

NOTE: Only one joystick can be connected to either one of the connector.

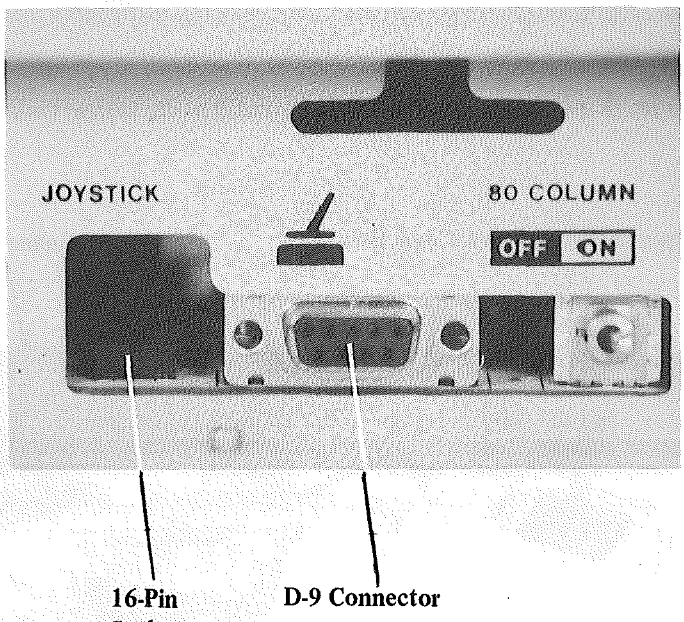


FIG 2–15 Two types of joystick connector

9 How to connect a Joystick

There are two main types of joystick connection commonly used. Your LINGO PC–128 mkII has both types of connectors available. The two types of connectors are located as shown.

NOTE: Only one joystick can be connected to either one of the connector.

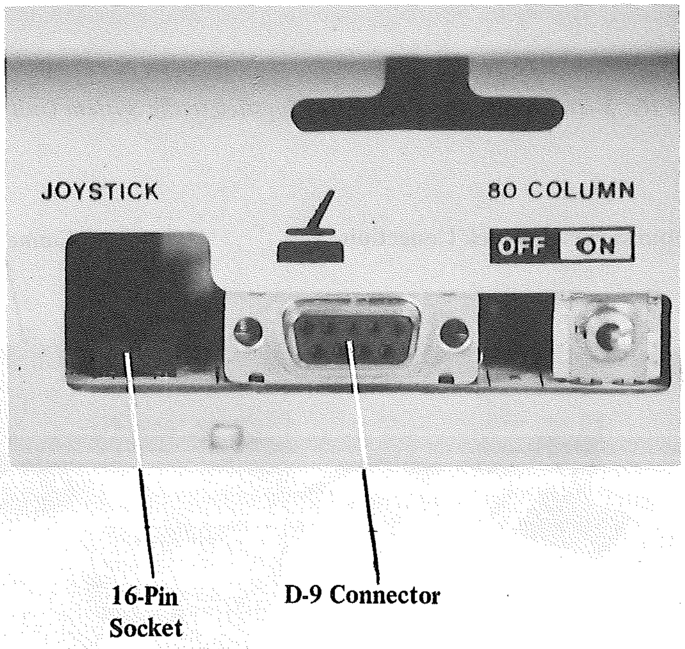


FIG 2–15 Two types of joystick connector

D–9 Joystick Connection.

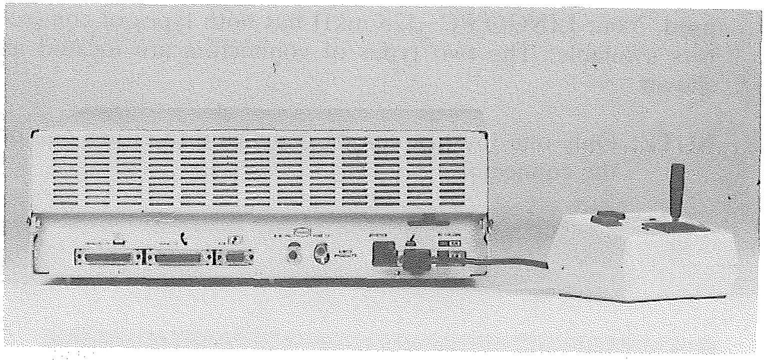


FIG. 2–16 Connecting a D–9 type joystick to the System Unit

16–pin socket Joystick Conection.

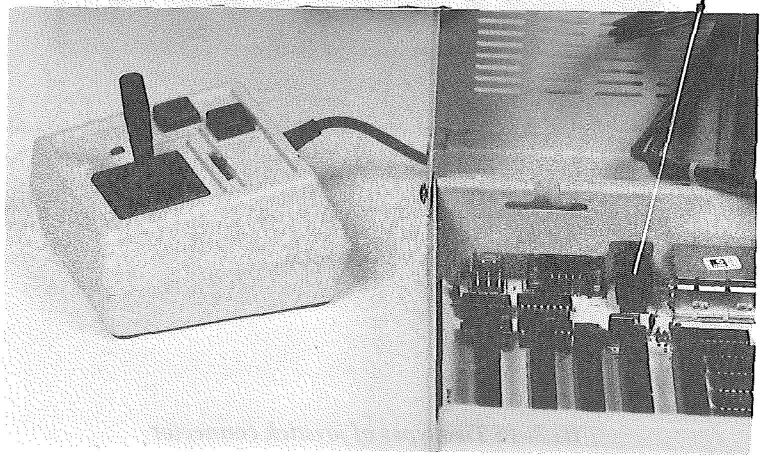


FIG. 2–17 Connecting a 16-pin type joystick to the System Unit

10 How to connect a Modem

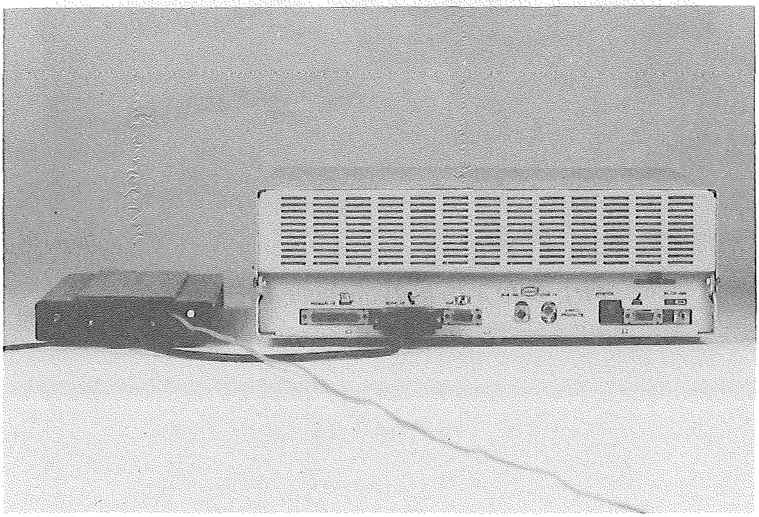
Locate the input connector of the modem labelled "RS 232C".

Locate the output connector of the System Unit labelled "SERIAL I/O".

Connect a modem cable as shown.

Connect a modular cable from the modem to the telephone line as shown.

Connect the modem power cord to the wall outlet.



Telephone
Wall Socket

FIG. 2–18 Connecting a modem to the System Unit

11 Component Arrangement

When cabling of your LINGO PC–128 mkII is completed, move your system components to the desired operating position as shown.

FIG. 2–19 A typical computer arrangement

The keyboard can be tilted for your typing comfort by lifting the "stands" located at the back of the keyboard.

CAUTION: Hold keyboard firmly when changing position.

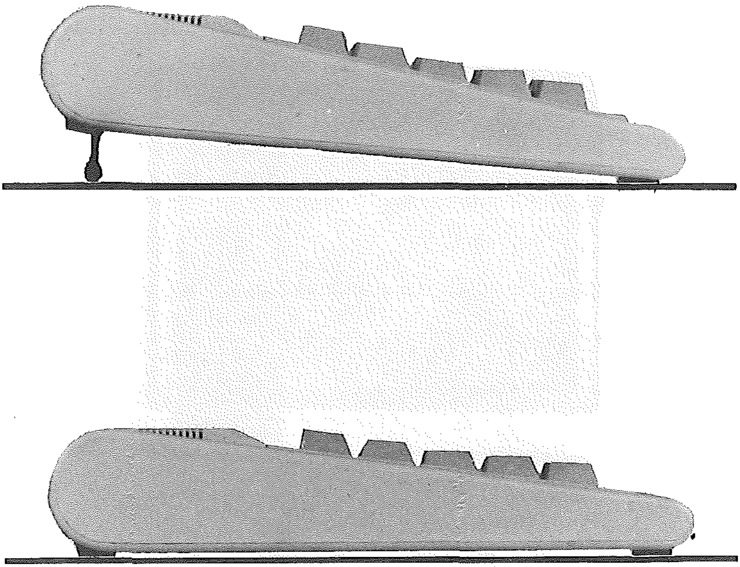


FIG. 2–20 A keyboard tilted in different ways

SWITCHING ON YOUR COMPUTER SYSTEM

Adjust your monitor or TV contrast and brightness to the maximum by turning the controls fully clockwise.

If you are using a TV, select an unused channel. Set the band select for that channel to UHF Channel 36.

Adjust the volume of your TV to minimum.

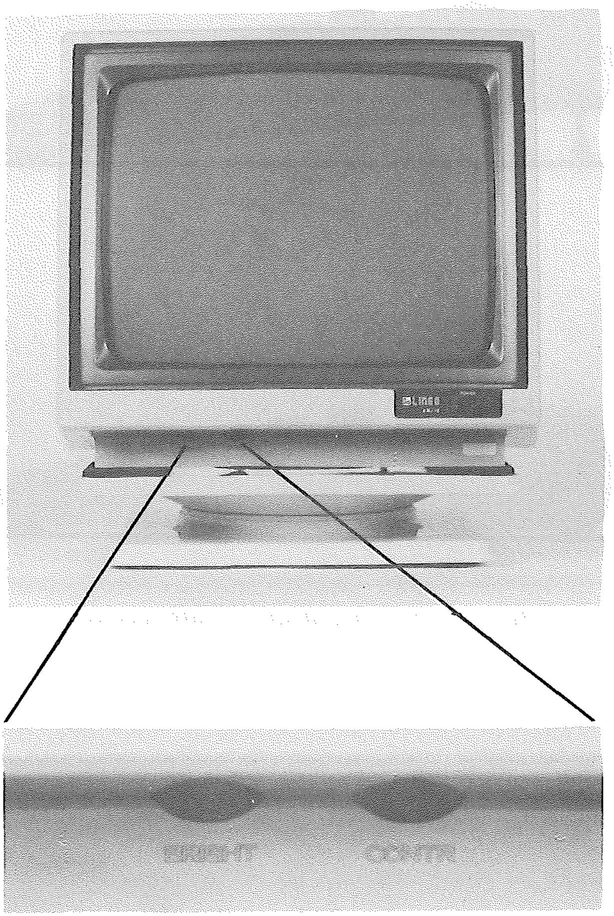


FIG. 2–21 Adjusting the contrast and brightness of the monitor

Switch on all your peripheral devices.

Then position the System Unit switch to ON. There will normally be three responses:

1. A short beep will be heard.
2. Disk Drive A "IN USE" light and power indicator light will be on.
3. If a Basic Rom card is not installed, then a LINGO logo should be displayed as shown.

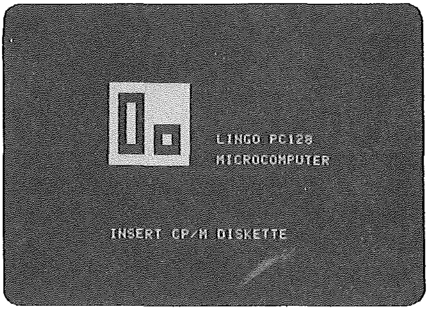
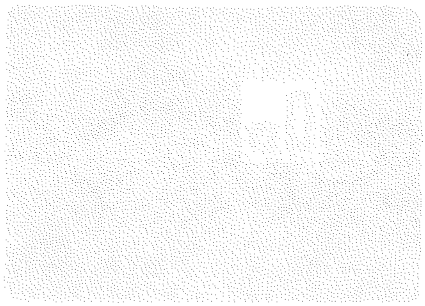


FIG. 2–22 A LINGO logo displayed on the screen

Adjust the contrast and brightness controls for eye comfort. If your are using a HOME TV, tune the TV channel slowly until a display appears.



SECTION 3 DISK AND DISK DRIVE

Contents **Page**

CHOOSING THE RIGHT DISKETTES

Your LINGO PC–128 mkII computer uses 5¼″ (133mm) single sided, single density as well as single sided, double density floppy diskettes for storing information (you may also have heard of the terms "disk" or "mini–floppy"). Each diskette can hold more than 143,000 characters.

ABOUT DISKETTES

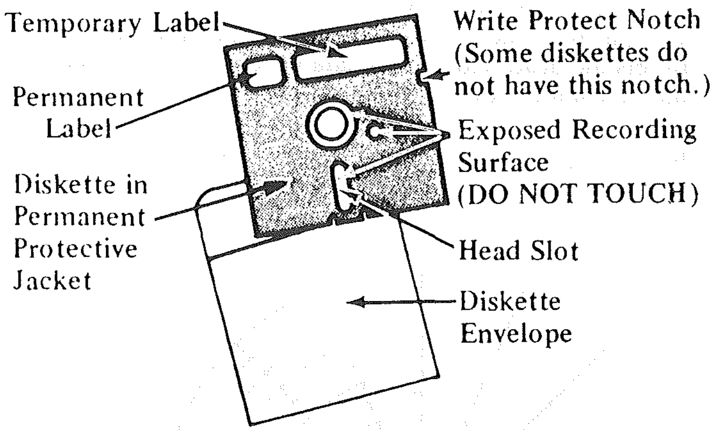


FIG. 3–1 A diskette

The permanent protective jacket (which is black) contains a flexible diskette which is coated with a magnetic substance. When in use, the diskette spins inside the jacket. The Read/Write head of the disk drive comes into contact with the recording surface through the long hole in the protecting jacket, called the head slot.

The information on a diskette can be read by the computer as often as it needs, or the computer can write new information on the diskette in an unused space.

The computer may also replace old information with new information by writing over it. In this case, the old information is erased and can no longer be read.

HOW DATA IS STORED ON A DISKETTE

Information is written on the diskette along concentric circles called tracks. The Read/Write head of the disk drive moves back and forth from one track to another as the diskette spins over it. This lets the head find certain data to read or find a place to write some new information.

There are 40 tracks on a diskette, numbered 0 to 39. Each track is divided into 16 sectors, and each sector can store 256 bytes of information.

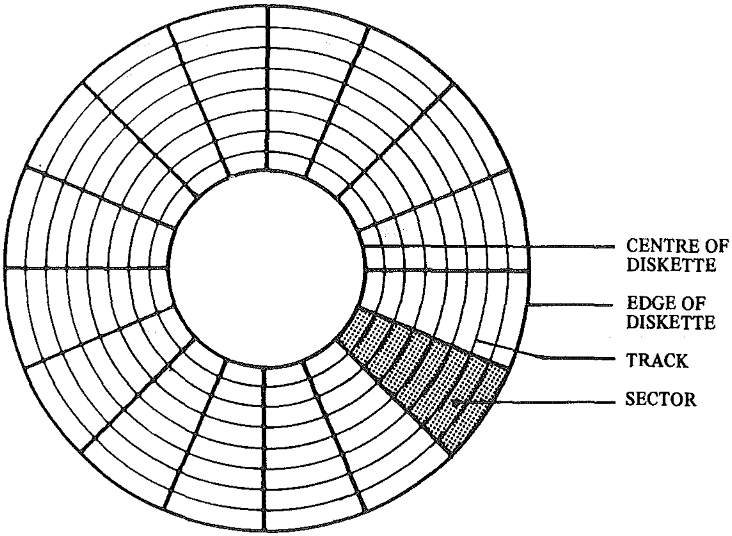


FIG. 3–2 Diskette Tracks and Sectors

WRITE–PROTECTING DISKETTE

Diskette can be physically protected against accidental erasure or overwriting. To protect the diskette against being written by the drive, cover the notch with a small label or tab. This notch is called a write–protect notch.

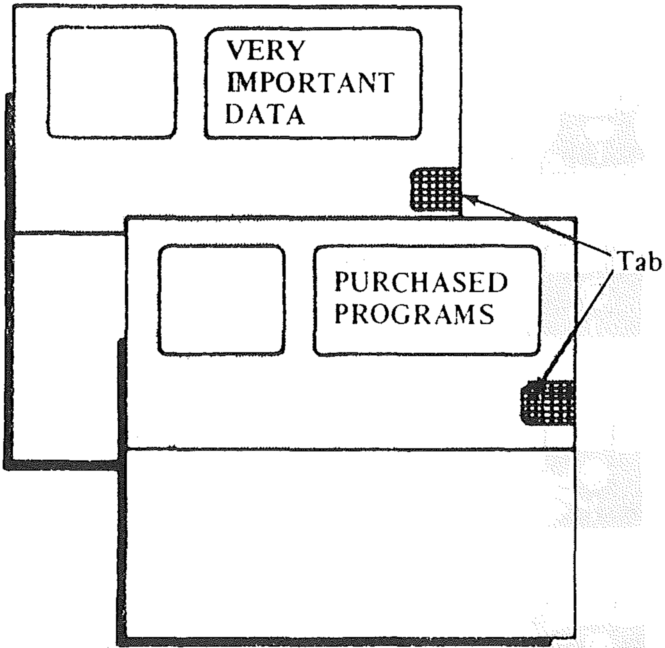
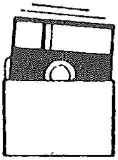


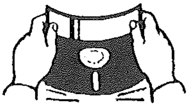
FIG. 3–3 Write–protecting a diskette

If you are certain it is okay to write on a write–protected diskette, you can peel off the tab so that the computer can write on it.

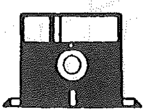
DISKETTE CARE



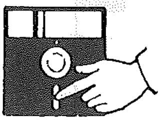
1. Protect the diskettes from dust by returning them to their envelopes as soon as you remove them from the disk drive.



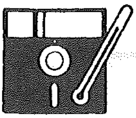
2. Do not bend your diskette



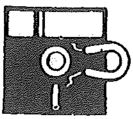
3. Always insert diskette gently into drive.



4. Do not touch the exposed recording surfaces.



5. Store seldom-used diskettes in storage boxes away from heat.



6. Keep diskettes away from sources of magnetic fields.

HOW TO INSERT A DISKETTE INTO THE DRIVE

a) If the drive door is already closed, press the lever of the drive door down and release.

b) Insert the diskette gently with the label upwards.

c) Push the diskette gently until the diskette is entirely in the drive.

d) Close the door of the disk drive by pressing the lever down until a click sound is heard.

Diskette

FIG. 3–4 Inserting a diskette into the drive

HOW TO REMOVE A DISKETTE FROM THE DRIVE

a) Press down the lever of the drive door until a click sound is heard.

b) Release the lever and the diskette will eject automatically.

c) Slowly slide the diskette out of the drive and replace it back into its envelope.

PREVENTING DATA LOSS

The following points will help prevent data being erased.

a) Wait until the "IN USE" light of the drive is off before switching off the System Unit.

b) Switch on the System Unit before inserting diskette.

c) Remove the diskette from the drive before switching off the System Unit.

WARNING: SWITCHING OFF THE COMPUTER WHILE THE DRIVE "IN USE" LIGHT IS ON MAY PERMANENTLY DESTROY THE DATA ON YOUR DISKETTE.

SECTION 4 KEYBOARD

Contents **Page**

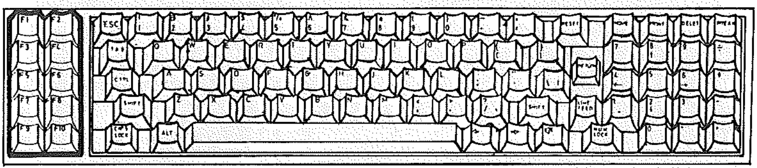
FIG. 4–1 PC–128 mkII Keyboard

The PC–128 mkII keyboard consists of 90 keys and two LED indicators indicating the status of CAPS LOCK and NUM LOCK. The keyboard is connected to the system unit by a cable and connector. Most of the keys used are the same as a typewriter except for certain keys. The keyboard is of ergonomic design and easy to operate.

All keys are auto–repeat. The keyboard converts the key position information to ASCII (American Standard Code for Information Interchange) codes. It can generate 128 different ASCII codes.

FIXED FUNCTION KEYS

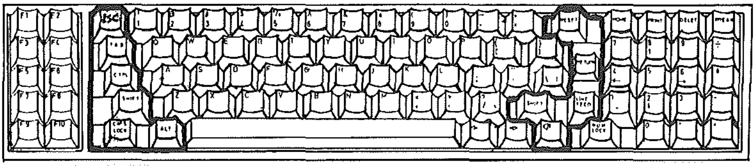
On the left side of the keyboard, you will find 10 fixed function keys numbered **F1–F10** The odd numbered keys are **CP/M** functions and the even numbered keys are **DOS** functions. Functions are performed for each key stroke by pressing the key itself.



Key	Description
F1	DIR A: displays the disk directory on drive A:
F2	CATALOG,D1 displays the disk directory on drive A:
F3	DIR B: displays the disk directory on drive B:
F4	CATALOG,D2 displays the disk directory on drive B:
F5	STAT *.* provides statistics on the file size and remaining space
F6	PR#6 activates the drive
F7	COPY B:=A: copies from drive A: to B:
F8	CALL–151 jumps from BASIC to MONITOR mode
F9	FORMAT format diskette on assigned drive
F10	INIT HELLO initialise diskette on assigned drive

Special Function keys

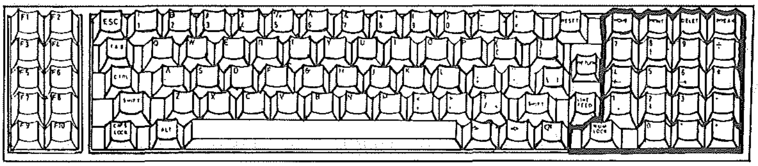
Some of the key works in conjunction with another key to perform a function. Functions are performed for each key stroke.



Key	Description
ALT	This key will toggle the fixed function key to redefinable function keys.
CTRL	This key is used with another key to perform a command or function. E.g. CTRL – RESET will cause the computer to stop whatever it is doing and reboot.
CAPS LOCK	This key is used to lock the alpha–characters in upper–case.
ESC	This key has a special function depending on the software used.
LINE FEED	Moves the cursor one line down.
RETRN	This key moves the cursor from the last character on one line to the first character of the next line. It indicates the logical end of a line.
RESET	This key is used with the CTRL key to obtain warm start or to stop program while it is running.
SHIFT	This key changes lower case letters to capitals.
TAB	This key performs a tab function similar to a typewriter.
	This key is used to switch the keyboard speaker on or off.

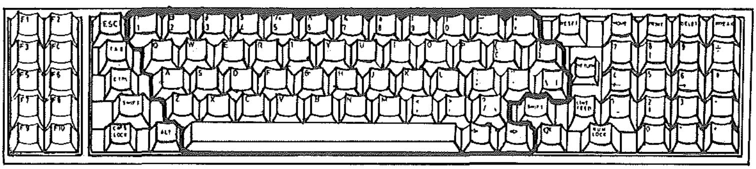
Numeric Key Pad

On the right side of the keyboad, you will find a numeric key pad. Press NUM LOCK key (the lamp lights up) and you have the 10–key numeric pad as found on a calculator for input of numeric data. To use the cursor movement key, switch NUM LOCK to "OFF" position (the light goes off).



Typewriter Key Pad

In the center of the keyboard, you will find 26 letter keys, 10 number keys and a total of 32 special symbol keys. On the bottom of the keyboard you will find a long bar, which is the space bar. These keys are similar to those of the typewriter.



LINE EDITING COMMANDS

CP/M Version 2.2

The screen display is automatically switched to 80 column display after booting. CP/M accepts commands both in uppercase and lowercase character.

CTRL–C	Reboots CP/M (warm boot).
→	Moves the cursor to the next tab stop. Tab stops are automatically set at each eighth column. Has the same effect as CTRL–U.
CTRL–E	Moves the cursor to the beginning of the next line without erasing the previous input.
DELETE	Deletes a character.
←	Moves the cursor to the left and removes one character for each key stroke.
CTRL–M	Sends the command line to CP/M and returns the cursor to the beginning of a new line. Has the same effect as RETURN.
CTRL–P	Copy all subsequent console output to the currently assigned list device. Output is sent to both the list device and the console device until the next CTRL–P is pressed.
CTRL–Q	Restarts screen scrolling after a CTRL–S.
CTRL–R	Places a # at the current cursor location, moves the cursor to the next line, and retypes any partial command you typed so far.
CTRL–S	Temporarily stops screen scrolling or printing.
CTRL–X	Discard all characters left of the cursor and moves the cursor to the beginning of the current lines. Save any characters right of the cursor.

CTRL–A	Non–destructive backspace.
CTRL–B	Moves the cursor to the beginning of the command line without having any effect on the contents of the line. If the cursor is at the beginning, CTRL–B moves it to the end of the line.
CTRL–G	Deletes the character indicated by the cursor. The cursor does not move.
CTRL–I	Moves the cursor to the next tab stop. Tab stops are automatically set at each eighth column.
CTRL–K	Deletes to the end of the line from the cursor.
CTRL–W	Recall and display previously entered command line both at the operating system level and within executing programs. If you press RETRN, CP/M 3.0 will execute the recalled command.

DOS Version 3.3

NOTE: The following functions are only available if the Basic Rom Card is installed.

Functions are performed for each key stroke.

→	Moves the cursor to the right and copies one character for each key stroke.
←	Moves the cursor to the left and removes one character for each key stroke.
BREAK	This key is used to stop program execution while it is running. Like a CTRL–C combination.
PRINT	This key is used in BASIC for print command.
CTRL–X	Ignored current displayed line and moves cursor to lower left margin.
CTRL–S	Stop or resume program listing.
ESC–A	Moves the cursor one character to the right. Press and release ESC before the key "A" is pressed and released.
ESC–B	Moves the cursor one character to the left.
ESC–C	Moves the cursor down one line.
ESC–D	Moves the cursor up one line.
ESC–E	Deletes all characters from the cursor to the end of the display line.
ESC–F	Deletes all characters from the cursor to the end of the display screen.

(Continue)

ESC–I	Press and release ESC to switch to editing mode. Press the key “I” to move the cursor up one line. This function is use to edit programs.
ESC–M	Moves the cursor down one line in editing mode.
ESC–J	Moves the cursor one character to the left in editing mode.
ESC K	Moves the cursor one character to the right in editing mode.
ESC–SHIFT–P	Clears the screen and moves the cursor to the upper left corner.

80 COLUMN DISPLAY

To obtain 80 column display, execute the command **PR#3**

CTRL–G	Sound the bell. The bell will have a different sound when 80 column is activated.
CTRL–H	Non–destructive back–space function. Performs the same function as the left arrow.
CTRL–J	Line feed.
CTRL–K	Clear to the end of screen.
CTRL–L	Form feed. This will clear the screen and home the cursor. The "HOME" in Applesoft program must be replaced with "PRINT CHR$(12)".
CTRL–M	Carriage return.
CTRL–N	Set inverse video off.
CTRL–O	Set inverse video on.
CTRL–S	Stop/Start text listing.
CTRL–U	Moves the cursor to the right and copies one character for each key stroke.
CTRL–Y	Home the cursor.
CTRL–Z	Typing CTRL–Z followed by: '3' set inverse video on '2' set inverse video off
CTRL–SHIFT–M	Clear to the end of line.
CTRL–SHIFT–N ‹x› ‹y›	Cursor position. The position of the cursor is determined by the ASCII values x and y. E.g., CTRL–SHIFT–N ‹space› ‹space› will position the cursor at the upper left corner of the screen.

Cursor movement keys are the same as in DOS Version 3.3.

How to install Wordstar

The video display mode of PC–128 has similar features as the Videoterm 80 Column Hardware card. If it is installed in WORDSTAR, user can use the 'SHIFT' key instead of the 'ESC' key to control the upper and lower case mode.

a) Boot up the Wordstar and execute the installation program.

A > INSTALL

b) Type 'Y' to the question – Do you want a normal first time installation of wordstar?. This will install the WSU.COM (uninstalled Wordstar) on the current drive and save it in the WS.COM (Wordstar) file.

c) Choose 'D' to modify the WS.COM file. This will replace the original modification.

d) Type 'WS' in order to state which file you would like to install.

e) Now you are in the 'Apple Video Board Menu'. Choose 1, Videoterm Hardware U/L Case. This helps to install the hardware 'Shift key'. To ensure, type 'Y'.

f) Type 'U' to unchange the rest of the menu.

g) Type 'Y' to confirm the Installation. This complete the procedure of installation.

SECTION 5 INTRODUCTION TO CP/M 3

Contents **Page**

INTRODUCTION

This section explains what CP/M does for you, how to startup, how to enter and edit the commands and how to make back–up copies of your CP/M diskettes.

Your LINGO PC–128 mkII is able to operate under various versions of CP/M Operating Systems such as CP/M VER. 2.0, VER. 2.2, and VER. 3.0. Our discussion will be directed towards CP/M 3.0.

WHAT IS CP/M?

Disk Operating System or DOS is an organized group of procedures and techniques that manages the overall operation of the computer.

CP/M is a disk operating system designed for Z−80 and 8080 microprocessor–based computer systems. CP/M stands for "Control Program for Microprocessor" and is produced by a company called Digital Research.

WHAT CP/M 3 DOES FOR YOU

CP/M 3 manages and supervises your computer's resources, including memory and disk storage, the console (screen and keyboard), printer, and communication devices. It also manages information stored on disks and grouping this information into files of programs or data. CP/M 3 can copy files from a disk to your computer's memory, or to a peripheral device such as a printer. To do this, CP/M 3 places various programs in memory and executes them in response to commands you enter at your console.

Once in memory, a program executes through a set of steps that instruct your computer to perform a certain task. You can use CP/M 3 to create your own programs, or choose from the wide variety of CP/M application programs that entertain you, educate you, help you solve commercial and scientific problems.

HOW TO START CP/M 3

Starting or loading CP/M 3 means reading a copy of the operating system from your CP/M system disk into your computer's memory. This process is also commonly referred to as 'booting'.

1) Position your System Unit to ON.

2) Insert the CP/M 3 System disk into Drive A.

3) Close the drive door.

After CP/M 3 is loaded into memory, a message similar to the following is displayed on your screen:

```
CP/M V3.0 Loader
Copyright (C) 1982, Digital Research

53K TPA

Installed for
LINGO PC-128 mkII Microcomputer
```

After this display, the following two-character message appears on your screen:

```
A>
```

This is the CP/M 3 system prompt. The system prompt tells you that CP/M 3 is ready to read a command from your keyboard. In this example, the prompt also tells you that drive A is your default drive.

WHY YOU SHOULD BACK UP YOUR FILES

Humans have faults, and so do computers. Human or computer errors sometimes destroy valuable programs or data files. By mistyping a command, for example, you could accidently erase a program that you just created or a data file that has been months in the making.

To avoid losing programs and data, makes copies of valuable files. Always make a working copy of any new program that you purchase and save the original. If the program is accidentally erased from the working copy, you can easily restore it from the original.

So far, we have not discussed any commands that change information recorded on your CP/M 3 system disk. Before we do, make a few copies of your system disk.

HOW TO MAKE COPIES OF YOUR CP/M 3 SYSTEM DISK

To back up your CP/M 3 disks, you need two or more diskettes. This section also shows how to make system disk back–ups on your system that has two drives: drive A and B.

Make sure that your system disk is in drive A, the default drive, and the blank disk in drive B. Follow the following procedures.

1) FORMATTING NEW DISKETTE

New diskette needs to be formatted before it could be used. This process prepares the disk in a predetermined arrangement so that it can store data. This is accomplised by the disk formatting program, FORMAT.COM, on your system disk.

SCREEN DISPLAY	THE ACTIONS THAT YOU RESPOND
A >	Type FORMAT
A > FORMAT	
FORMAT Vers 1.0	
40–Tracks–Drve Formatter	
Format which disk?	Type B
Format 40 Tracks?	Type Y
Press <RETURN> to begin	Type Y to format another Type N to abort

2) COPYING DISKETTES

Executing the disk copying program, COPY.COM, on your system disk will make a duplicate copy of your original disk.

SCREEN DISPLAY	THE ACTIONS THAT YOU RESPOND
A >	Type COPY
A > COPY	
COPY Vers. 1.0 5″ Disk Duplication	
Source disk?	Type A.
Destination disk?	Type B.
Copy 40 Tracks ?	Type Y.
Press <RETURN> to begin	Press RETRN key.
Copying...	
COPY another disk (Y/N)?	Type Y to copy another. Type N to abort.

SECTION 6 CP/M 3 COMMANDS

Contents **Page**

THE COMMAND LINE

CP/M 3 performs tasks according to specific commands that you type at your keyboard. A CP/M 3 command line is composed of a command keyword, an optional command tail, and a carriage return keystroke. The command keyword identifies a command (or program) to be executed. The command tail can contain extra information for the command, such as a filename or parameters. To end the command line, you must press the carriage return (RETRN) key. The following example shows a user entering a command line.

A > DIR FORMAT.COM

In this example, DIR is the command keyword and FORMAT .COM is the command tail. The carriage return keystroke does not appear on the screen or in the example. You must remember to press the carriage return key to send a command line to CP/M 3 for processing. Note that the carriage return key is marked RETRN on your keyboard.

As you type characters at the keyboard, they appear on your screen. The single−character position indicator, called the cursor, moves to the right as you type characters. If you make a typing error, press either the left arrow key or CTRL−H to move the cursor to the left, and correct the error. CTRL is the abbreviation for the Control key. To type a control character, hold down the control key and press the required letter key. For example, to move the cursor to the left, hold down CTRL key and press the H key.

You can type the keyword and command tail in any combination of upper–case and lower–case letters. CP/M 3 treats all letters in the command line as upper–case.

Generally, you type a command line directly after the system prompt. However, CP/M 3 does allow spaces between the prompt and the command keyword.

TYPES OF COMMANDS

CP/M 3 recognizes two different types of commands: built–in commands and transient utility commands. Built–in commands can be executed immediately. Transient utility commands are stored on disk as program files. They must be loaded from disk to perform their task. You can recognize transient utility programs files when a directory is displayed on the screen because their filenames are followed by COM.

BUILT–IN COMMANDS

Built–in commands are part of the CP/M 3 and are always available for your use regardless of which disk you have in which drive. Built–in commands resides in memory as a part of CP/M 3 and therefore execute more quickly than the transient utilities.

Below is a table of built–in commands with a short description of each.

Command	Function
DIR	Displays filenames of all files in the directory except those marked with SYS attribute.
DIRSYS	Displays filenames of files marked with the SYS (system) attribute in the directory.
ERASE	Erases a filename from the disk directory and release the storage space occupied by the file.
RENAME	Renames a disk file.
TYPE	Displays contents of an ASCII (TEXT) file at your screen.
USER	Changes to a different user number.

CP/M 3 allows you to abbreviate the built–in commands as follows:

DIRSYS .... DIRS
ERASE .... ERA
RENAME .... REN
TYPE .... TYP
USER .... USE

Let's use one command to demonstrate how CP/M 3 reads command lines. The DIR command, which is an abbreviation for directory, tells CP/M 3 to display a directory of disk files on your screen. Type the DIR keyword after the system prompt, omit the command tail, and press RETRN.

A > DIR

CP/M 3 responds to this command by displaying the names of all the files that are stored on the disk in drive A. For example, if you have your CP/M 3 system disk in drive A, the following will appear on your screen:–

```
A>DIR
A: CPM3    SYS : COPYSYS  COM : DEVICE  COM : DIR    COM : DUMP   COM
A: ERASE   COM : FORMAT   COM : GET     COM : PATCH  COM : PIP    COM
A: PUT     COM : RENAME   COM : SAVE    COM : SET    COM : SETDEF COM
A: SHOW    COM : SUBMIT   COM : TYPE    COM :
```

CP/M 3 recognizes only correctly spelled command If you make a typing error and press RETRN before correcting your mistake, CP/M 3 echoes the command line followed with a question mark. If you mistype the DIR command, as in the following example, CP/M 3 responds

A > DJR
DJR?

to tell you that it cannot find the command keyword. To correct simple typing errors, use the left arrow key, CP/M 3 supports other control characters that help you efficiently edit command lines.

DIR accepts a filename as a command tail. You can use DIR with a filename to see if a specific file is on the disk. For example, to check that the transient utiltiy program COPYSYS.COM is on your system disk, type

A > DIR COPYSYS.COM

CP/M 3 performs this task by displaying either a name of the file you specified, or the message, No File.

Be sure you type at least one space after DIR to separate the command keyword from the command tail. If you do not, CP/M 3 responds as follows.

```
A>DIRCOPYSYS.COM
DIRCOPYSYS.COM?
```

TRANSIENT UTILITY COMMANDS

When you enter a command keyword that identifies a transient utility, CP/M 3 loads the program file from the disk and passes it to any filenames, data, or parameters you entered in the command tail. Below is a table of transient utility commands with a short description of each.

Name	Function
COPY	Copies a programmed disk to another disk
COPYSYS	Creates a new boot disk.
DEVICE	Assign logical CP/M devices to one or more physical devices, changes device driver protocol or sets console screen size.
DUMP	Displays a file in ASCII and hexadecimal format.
GET	Temporarily gets console input from a disk file rather than keyboard.
PIP	Copies files and combines files.
PUT	Temporarily directs printer or console output to a disk file.

(Continue)

Name	Function
SET	Sets file options including disk labels, file attributes, type of time and date stamping, and password protection.
SETDEF	Sets the system options including the drive search chain.
SETUP	Set Console Input/Output and Peripheral Input/Output.
SHOW	Displays disk and drive statistics.
SUBMIT	Automatically executes multiple commands.

Each of the transient utility commands perform a number of specific tasks according to the command tail attached to it. Refer to Section 9 on 'COMMAND SUMMARY' for detailed description of each command.

SECTION 7 CP/M 3 FILES

Contents **Page**

INTRODUCTION

CP/M 3's most important task is to access and maintain files on your diskette. With CP/M 3 you can create, read, write, copy, and erase diskette files. This section tells you what a file is, how to create, name, and access a file, and how files are stored on your diskette. It also tells how to change diskette and change the default drive.

WHAT IS A FILE?

A CP/M 3 file is a collection of related information stored on a diskette. Every file must have a unique name because CP/M 3 uses that name to access the file. A directory is also stored on each disk. The directory contains a list of the filenames stored on that disk and the locations of each file on the disk.

In general, there are two kinds of files: program (command) files and data files. A program file contains an executable program, a series of instructions that computer follow step–by–step. A data file is usually a collection of information: a list of name and addresses, the inventory of a store, the accounting records of a business, the text of a document, or similar related information. For example, your computer cannot execute names and addresses, but it can execute a program that print names and addresses on mailing labels.

A data file can also contain the source code for a program. Generally, a program source file must be processed by an assembler or compiler before it becomes a program file. In most cases, an executing program processes a data file. However, there are times when an executing program processes a program file. For example, the copy program PIP can copy one or more program files.

HOW ARE FILES CREATED?

There are many ways to create a file. One way is to use a text editor. You can also create a file by copying an existing file to a new location, perhaps renaming it in the process. Under CP/M 3, you can use the PIP command to copy and renames files.

HOW ARE FILES NAMED?

CP/M 3 identifies every file by its unique file specification. A file specification can be simply one to eight character filename, such as:

MYFILE

A file specification can have four parts: a drive specifier, a filename, a filetype, and a password.

The drive specifier is a single letter (A–P) followed by a colon. Each drive in your system is assigned a letter. When you include a drive specifier as part of the file specification, you are telling CP/M 3 that the file is stored on the disk currently in that drive. For example, if you enter

B:MYFILE

CP/M 3 looks in drive B for the file MYFILE.

The filename can be from one to eight characters. When you make up a filename, try to let the name tell you something about what the file contains. For example, if you have a list of customer names for your business, you could name the file:

CUSTOMER

so that the name gives you some idea of what is in the file.

As you begin to use your computer with CP/M 3, you will find that files fall naturally into categories. To help you identify files belonging to the same category, CP/M 3 allows you to add an optional one–to–three–character extension, called a filetype, to the filename. When you add a filetype to the filename, separate the filetype from the filename with a period. Try to use three letters that tell something about the file's category. For example, you could add the following filetype to the file that contains a list of customer names:

CUSTOMER.NAM

When CP/M 3 displays files specifications in response to a DIR command, it adds blanks to short filenames so that you can compare filetypes quickly. The program files that CP/M 3 loads into memory from a disk have different filenames, but all have the filetype COM.

With LINGO PC–128 mkII, you can add a password as an optional part of the file specification. The password can be from one to eight characters. If you include a password, separate it from the filetype(or filename, if no filetype is included) with a semicolon follows:

CUSTOMER.NAM;ACCOUNT

If a file has been protected with a password, you must ENTER the password as part of the file specification to access the file.

We recommend that you create filenames, filetypes, and passwords from letters and numbers. You must not use the following characters in filenames, filetypes, or passwords because they have special meanings for CP/M 3:

‹ › = , ! ! * ? & / $ [ ] ( ) . : ; + –

A complete file specification containing all possible elements consists of a drive specification; a primary filename, a filetype and a password, all separated by their appropriate delimiters, is shown the following example:

A:DOCUMENT.LAW;SUSAN

drive specifier | filename | filetype | password

CP/M 3 has already establised several file groups. The following table lists some fo their filetype with a short description of each family.

Filetype	Meaning
ASM	ASseMbler source file
BAS	BASic program file
COM	COMmand program file
HLP	HELP message file
SUB	List of commands to be executed by SUBMIT
TXT	TeXT file
BAK	BAcKup file
$$$	Temporary file

DO YOU HAVE THE CORRECT DRIVE?

When you type a file specification in a command tail without drive specifier, the program looks for the file in the drive named by the system prompt, called the default drive. For example, if you type the command as follow:

A > DIR COPYSYS.COM

DIR looks in the directory of the disk in drive A for COPYSY-S.COM. If you need to access the disk in drive B, you should precede a filename with a drive specifier. For example, in response to the command

A > DIR B:DBASE.COM

CP/M 3 looks for the file DBASE.COM in the directory of the disk
in drive B. When you give a command to CP/M 3, note which disk is in the default drive. Many application programs require that the data files they access be stored in the default drive.

You can also precede a program filename with a drive specifier, even if you use the program filename as a command keyword. For example, if you type the following command:

A > B:PIP

CP/M 3 looks in the directory of the disk in drive B for the file PIP.COM. If CP/M 3 finds PIP on drive B, it loads PIP into memory and executes it.

If you need to access many files on the same drive, you might find it convenient to change the default drive so that you do not need to repeatedly enter a drive specifier. To change the default drive, enter the drive specifier next to the system prompt and press RETRN. In response, CP/M 3 changes the system prompt to display the new default drive:

A > B:
B >

Unlike the filename and filetype which are stored in the disk directory, the drive specifier for a file changes as you move the disk from one drive to another. Therefore, a file has a different file specification when you move a disk from one drive to another.

USER NUMBER

CP/M 3 further identifies all files by assigning each one a user number which ranges from 0 to 15. CP/M 3 assigns the user number to a file when the file is created. User numbers allow you to separate your files into sixteen file groups. User numbers are particularly useful for organizing files on a hard disk.

When you use CP/M 3 utility to create a file, the file is assigned to the current user number, unless you use PIP to copy the file to another user number. You can determine the current user number by looking at the system prompt.

4A >	User number 4, drive A
A >	User number 0, drive A
2B >	User number 2, drive B

The user number always precedes the drive specifier. User 0, however, is the default user number and is not displayed in the prompt.

You can use the built-in command USER to change the current user number.

```
A > USER 3
3A>
```

You can change both the user number and the drive by entering the new user number and drive specifier together at the system prompt:

```
A > 3B:
3B >
```

Most commands can access only those files that have the current user number. For example, if the current user number is 7, a DIR command with no options displays only the files that were created under user number 7.

```
A > 7B
7B > DIR
No file
7B >
```

ACCESSING MORE THAN ONE FILE

Certain CP/M 3 built–in and transient utilities can select and process several files when special wildcard characters are included in the filename or filetype. A file specification containing wildcard is called an ambiguous filespec and can refer to more than one file because it gives CP/M 3 a pattern to match. CP/M 3 searches the disk directory and selects any file whose filename or filetype matches the pattern.

The two wildcard characters are?, which matches any single letter in the same position, and *, which matches any character at that position, and any other characters remaining in the filename or filetype. The following list presents the rules for using wildcards.

– A? matches any character in a name, including space character.

– An * must be the last, or only, character in the filename or filetype. CP/M 3 internally replaces an * with? characters to the end of the filename or filetype.

– When the filename to match is shorter than eight characters, CP/M 3 treats the name as if it ends with spaces.

– When the filetype to match is shorter than three characters, CP/M 3 treats the filetype as if it ends with spaces.

Suppose, for example, you have a disk that contains the following six files:

A.COM, AA.COM, AAA.COM, B.COM,
A.ASM, and B.ASM

The following wildcard specification match all, or a portion of, these files:

* *	is treated as ????????.???
????????.???	matches all six names
*.COM	is treated as ????????.COM
????????.COM	matches the first four names

?.COM	matches A.COM AND B.COM
?.*	is treated as ?.???
?.???	matches A.COM, B.COM, A.ASM and B.ASM
A?.COM	matches A.COM and AA.COM
A*.COM	is treated as A???????.COM
A???????.COM	matches A.COM, AA.COM, and AAA.COM

Remember that CP/M 3 uses wildcard pattern only while searching a disk directory, and therefore wildcards are valid only in filename and filetypes. You cannot use a wildcard character in a drive specifier. You also cannot use a wildcard character as part of a filename or filetype when you create a file.

Suppose, for example, you want to have a directory listing of all the .COM files in drive A, you can make use of the "*" wildcard function.

```
A>DIR A:*.COM
```

This command will list all the files in drive A with the filetype .COM.

HOW TO PROTECT YOUR FILES

Under CP/M 3 you can organized your files into groups to protect them from accidental change and from unauthorized access. You can specify how your files are displayed in response to a DIR command. CP/M 3 supports these features by assigning the following to files:

- user numbers
- attributes
- password

All of this information for each file is recorded in the disk directory.

FILE ATTRIBUTES

File attributes control how a file can be accessed. When you create a file, CP/M 3 gives it two attributes. You can change the attributes with a SET command.

The first attribute can be set to either DIR (Directory) or SYS (System). This attribute controls whether CP/M 3 displays the file's name in response to DIR command or DIRSYS command. When you create a file, CP/M 3 automatically sets this attribute to DIR. You can display the name of the file marked with the DIR attribute, with a DIR command. If you give a file the SYS attribute, you must use a DIRSYS command to display the filename. Simple DIR and DIRSYS commands display only the filenames created under the current user number.
A file with the SYS attribute has a special advantage when it is created under user 0. When you give a file with user number 0 the SYS attribute, you can read and execute that file from any user number. This feature gives you a convenient way to make your commonly used programs available under any user number.

The second file attribute can be set to either R/W (Read–Write) or R/O (Read–Only). If a file is marked R/O, any attempt to write data to that file produces a Read–Only error message. Therefore, you can use the R/O attribute to protect important files. A file with the R/W attribute can be read or written to, or erased at any time, unless the disk is physically write–protected.

PASSWORD

Passwords allow you to protect your files from access by other users. You can use passwords to limit access to certain files for security purposes.

The SET utility allows you to enable password protection on a drive, assign a password to SET itself (so that unathorized users cannot disable password protection on a drive), and assign passwords to specific files that have already been created. You can assign passwords to all program and data files. This means that a command line could require the entry of two passwords in order to execute: one password to access to the command program, and a second password to access to the file specified in the command tail. Refer to the section on SET command for more details.

Some CP/M 3 commands and most word processing, accounting and other application programs running under CP/M 3 do not accept passwords in the command tail. If you want to protect your file and still use those program, you can set a default password before executing the application program.

HOW ARE FILES STORED ON A DISK?

CP/M 3 records the filename, filetype, password, user number, and attributes of each file in a special area of the disk called the directory. In the directory, CP/M 3 also records which parts of the disk belong to which file.

CP/M 3 allocates directory and storage space for a file as records are added to the file. When you erase a file, CP/M 3 reclaims storage in two ways: it makes the file's directory space available to catalog a different file, and frees the file's storage space for later use. It is this dynamic allocation feature that makes CP/M 3 powerful. You do not have to tell CP/M 3 how big your file will become, because it automatically allocates more storage for a file as needed, and releases the storage for reallocation when the file is erased. Use the SHOW command to determine how much space remains on the disk.

```
A > SHOW A:
A:RW, Space : 31K
```

CHANGING FLOPPY DISKETTE

CP/M 3 cannot, of course, do anything to a file unless the disk that holds the file is inserted into a drive and the drive is ready. When a disk is in a drive, it is online and CP/M 3 can access its directory and files.

At some time, you will need to take a disk out of a drive and insert another that contains different files. You can replace an online disk whenever you see the system prompt at your console. This is a clear indication that no program is reading or writing to the drive.

You can also remove a disk and insert a new one when an application program prompts you to do so. This can occur, for example, when the data that the program uses does not fit on one floppy disk.

NOTE: You must never remove a disk if a program is reading or writing to it.

You can change disks on the drive without sending any special signal to CP/M 3. This allows you to insert another disk at a program's request and read files from or create files on the new disk.

NOTE: The older versions of CP/M requires you to type CTRL–C whenever you change disks.

PROTECTING A DRIVE

Under CP/M 3, drives can be marked R/O just as files can be given the R/O attribute. The default state of a drive is R/W. You can give a drive the R/O attribute by using the SET command. To return the drive to R/W, use the SET command or press a CTRL–C to the system prompt.

```
A > SET A: [RO]
```

The preceding command sets drive A to Read–Only access. All files in drive A can be read or executed, but not deleted.

```
A > SET A: [RW]
```

The preceding command sets drive A to Read–Write access. All files in drive A can be deleted.

SECTION 8 CONSOLE AND PRINTER

Contents **Page**

INTRODUCTION

This section describes how CP/M 3 communicates with your console (screen and keyboard) and printer. It tells how to start and stop console and printer output, edit commands you enter at your console, and redirect console and printer input and output. It also explains the concept of logical devices under CP/M 3.

CONTROLLING CONSOLE OUTPUT

Sometimes CP/M 3 displays information on your screen too quickly for you to read it. Sometimes an especially long display scrolls off the top of your screen before you have a chance to study it. To ask the system to wait while you read the display, hold down the CTRL key and press S. A CTRL–S keystroke causes the display to pause. When you are ready, press CTRL–Q to resume the display. If you press any key besides CTRL–Q during a display pause, CP/M beeps the speaker.

DIR, TYPE, and other CP/M utilities support automatic paging at the console. This means that if the program's output is longer than what the screen can display at a one time, the display automatically halts when the screen is filled. When this occurs, CP/M 3 prompts you to press RETRN to continue and the display continues.

CONTROLLING PRINTER OUTPUT

You can also use a control command to echo console output to the printer, that is to display on the screen as well as sending to the printer. To start printer echo, press a CTRL–P. To stop, press CTRL–P again. While printer echo is in effect, any characters that appear on your screen are listed at your printer.

You can use printer echo with a DIR command to make a list of files stored on a floppy disk. You can also use CTRL–P with CTRL–S and CTRL–Q to make a hard copy of part of a file. Use a TYPE command to start a display of the file at the console. When the display reaches the part you need to print, press CTRL–S to stop the display, CTRL–P to enable printer echo, and then CTRL–Q to resume the display and start printing. You can use another CTRL–S, CTRL–P, CTRL–Q sequence to terminate printer echo.

NOTE: The TYPE command is used to list text files only. For example:

```
A > TYPE HELP.DOC
            |
        Text File
```

CONSOLE LINE EDITING

You can correct simple typing errors with the left arrow key. CP/M 3 also supports additional line–editing with control characters. You can use the control characters to edit command lines or input lines to most programs.

Below is a table of line–editing control characters and a short description of each.

Character	Meaning
CTRL–B	Moves the cursor to the beginning of the command line without having any effect on the contents of the line. If the cursor is at the beginning, CTRL–B moves it to the end of the line.
CTRL–E	Forces a physical carriage return but does not send the command line to CP/M 3. Moves the cursor to the beginning of the next line without erasing the previous input.
CTRL–I	Moves the cursor to the next tab stop. Tab stops automatically set at each eight column.
CTRL–M	Sends the command line to CP/M 3 and returns the cursor to the beginning of a new line. Has the same effect as a RETRN or a CTRL–J keystroke.
←	Deletes a character and moves the cursor left one position.
→	Copy one character and moves the cursor right one one position.
CTRL–R	Retypes the command line. Places a # at the current cursor location, moves the cursor to the next line, and retypes any partial command you typed so far.
CTRL–W	Recall and displays previously entered command, line both at the operating system level without executing program, if the CTRL–W is the first character entered after the prompt. CTRL–J, CTRL–M, and RETRN define the command line you can recall. If the command line contains characters, CTRL–W moves the cursor to the end of the command line. If you press RETRN, CP/M 3 executes the recalled command.

(Continue)

Character	Meaning
CTRL–X	Discards all the characters left to the cursor and moves the cursor to the beginning of the current line. CTRL–X saves any characters right of the cursor.

The cursor control keys are supported by CP/M 3 . These are located at the numeric key pad. To use these keys, press the 'NUM LOCK' key so that the 'NUM LOCK' indicator light is off. A description of these cursor control keys are as follows:–

Character	Meaning
←	Moves the cursor one character to the left.
→	Moves the cursor one character to the right.
↑	Moves the cursor one line up.
↓	Moves the cursor one line down.
HOME	Erases the screen and moves the cursor to the top left–hand corner of the screen.

ASSIGNING LOGICAL DEVICES

Most CP/M 3 computer systems have a traditional console with a keyboard and screen display. Many also have letter–quality printers. If you use your computer for unusual tasks, you might want to add a different kind of character device to your system: a line printer, a teletype terminal, a modem, or even a joystick for playing games. To keep track of these physically different input and output devices, CP/M 3 associates the different physical devices with logic devices. Below is a table of the names of CP/M 3 logical devices. It also shows the physical devices assigned to these logical devices in the distributed CP/M 3 system. You can also change these assignments with a DEVICE comamnd.

Logical Device Name	Device Type	Physical Device Name
CONIN:	Console input(Keyboard)	CON108
CONOUT:	Console output(Screen)	CON108
AUXIN:	Auxiliary input	SERIAL
AUXOUT:	Auxiliary output	SERIAL
LST:	List output	SPOOL

SPOOLER

The spooler feature of CP/M 3 will enable you to print and input to the computer simultaneously. This will aid in cutting down waiting time. You can continue with your work on the computer, while the printer is still printing. CP/M 3.0 allocates 44K of memory space (Bank 1) to buffer the data to be printed. To interrupt and clear the buffer of any data, type ‹SHIFT›‹CTRL›‹F8›.

NOTE: Spooler is only supported on the parallel printer interface.

SECTION 9 COMMAND SUMMARY

Contents **Page**

INTRODUCTION

This section describes the commands and programs supplied with your CP/M 3 operating system. The commands are in alphabetical order. Each command is followed by a short explanation of its operation and examples. More complicated commands are described later in details.

HOW COMMANDS ARE DESCRIBED

Each command description is given in a specific form:

- The description begins with the commmand keyword in upper−case.

- The syntax section gives you one or more general forms to follow when you compose the command line.

- The explanation section defines the general use of the command keyword, and points out exceptions and special cases. The explanation sometimes includes tables and lists of options that you can use in the command line.

- The examples section lists a number of valid command lines that use the command keyword.

The notation in the syntax lines describes the general command form using these rules:

- Words in capital letters must be spelled as shown, but you can use any combination of upper− or lower− case letters.

- You must include one or more space characters where a space is shown, unless otherwise specified. For example, the PIP options do not need to be separated by spaces.

- The parts of a file specification are represented like this:

  d:filename.typ;password

  where d: represents the optional drive specifier, filename represents the one– to eight– character filename, and typ represents the optional one– to three–character filetype. The syntax descriptions in this section use the term filespec to indicate any valid combination of the elements included in the file specification.

The following tables defines the special symbols and abbreviations used in syntax lines.

Symbol	Meaning
DIR	Directory attribute
n	You can substitute a number for n.
o	Indicates an option or an option list
RO	Read–Only
RW	Read–Write
s	You can substitute a string, which consists of a group of charactrers, for s.
SYS	System attribute
{ }	Items within braces are optional. You can enter a command without the optional items. The optional items add effects to your command line.
[ ]	Items in square brackets are optional or an option list. If you use an option specified within the brackets, you must type the brackets to enclose the option. If the right bracket is the last character on the command line, it can be omitted.
( )	Items in parentheses indicate a range of options. If you use a range from an option list, you must enclose the range in parentheses.
...	Ellipses tell you that the previous item can be repeated any number of times.
|	The or bar separates alternative item in a command line. You can select any or all of the alternatives specified. Mutually exclusive options are indicated in additional syntax lines or are specifically noted in the text.
or CTRL	Represent the CTRL key on your keyboard.
< cr >	Represent the RETRN key on your keyboard.

(Continued)

Symbol	Meaning
*	Wildcard character -- replaces all or part of a filename and/or filetype.
?	Wildcard character -- replaces any single character in the same position of a filename or filetype.

Let's look at some examples of syntax notation. The CP/M 3 DIR(DIRectory) command displays the names of files cataloged in the disk directory and, optionally, displays other information about the files.

The syntax of the DIR command with options shows how to use the command line syntax notation:

```
Syntax:   DIR {d:}   {filespec}   {[options]}
                 |        |             |
          optional   optional      optional
```

This tells you that command tail following the command keyword DIR is optional. DIR alone is a valid command, but you can include a file specification, or a drive specification, or just the options in the command line. Therefore,

DIR
DIR filespec
DIR d:
DIR [RO]

are valid commands. Furthermore, the drive or file specification can be followed by another optional value selected from one of the following list of DIR options:

RO
RW
DIR
SYS

Therefore,

DIR d:filespec [RO]

is valid command.

Recall what you have learned about wildcards in filenames and filetypes. The DIR commands accepts wildcards in the file specification.

Using this syntax, you can construct several valid command lines:

```
DIR
DIR X.PAS
DIR X.PAS [RO]
DIR X.PAS [SYS]
DIR *.PAS
DIR *.* [RW]
DIR X.* [DIR]
```

The CP/M 3 command PIP (Peripheral Interchange Program) is the file copy program. PIP can copy information from the disk to the screen or printer. PIP can combine two or more files into one longer file. PIP can also rename files after copying them. Look at one of the formats of PIP command line for another example of how to use command line notation. PIP also copies disk to disk.

Syntax: PIP dest–filespec=src–filespec {,filespec...}

In the preceding example, dest–filespec is further defined as a destination file specification or peripheral device (printer, for example) that receives data. Similarly, src–filespec is a source file specification or peripheral device (keyboard, for example) that transmits data. PIP accepts wildcards in the filename and filetype. There are, of course, many valid command lines that come from this syntax. Some examples follow:

```
PIP NEWFILE.DAT=OLDFILE.DAT
PIP B:=A:THISFILE.DAT
PIP B:=X.BAS=Y.BAS, Z.BAS
PIP X.BAS=A.BAS, B.BAS, C.BAS
PIP B:=A:*.BAK
PIP B:=A:*.*
```

The remainder of this section contains a complete description of each CP/M 3 utility. The descriptions are arranged alphabetically for easy reference.

The COPY Command

Syntax: COPY

Explanation: The COPY command copies the CP/M programmed disk to another disk. The disk must have the same format as the original programmed disk. For example, if the programmed disk is a single–density disk, the disk you use to copy onto must also be in single–density format.

Example: A > COPY

COPY Ver. 1.0
5 ″ Disk Duplication

Source disk? A
Destination disk? B

Copy 40 tracks? Y

The preceding example copies the programmed disk from disk drive A to drive B.

The COPYSYS Command

Syntax: COPYSYS

Explanation: The COPYSYS command copies the CP/M 3 system from a CP/M 3 system disk to another disk. The disk must have the same format as the original system disk. For example, if the system disk is a single–density disk, the disk you use to copy onto must also be in single–density format.

The COPYSYS utility copies the system tracks as well as the system file CPM3.SYS to the new disk.

Example:

```
A > COPYSYS

System Writer Vers 1.0

Source Disk --------> ?A
Destination Disk ----- >?B

Press < RETURN > to begin

System Tracks Copied
System File Copied

Syswrt Finished
```

The preceding example copies the CP/M 3 system from disk drive A to drive B.

The DEVICE Command

Syntax: DEVICE {NAMES | VALUES | physical-dev logical-dev }
DEVICE logical–dev=physical–dev {options} {,physical–dev {options},...}
DEVICE logical–dev = NULL
DEVICE physical–dev {option}

DEVICE CONSOLE [PAGE|COLUMNS= columns|LINES=line]

Explanation: The DEVICE command is a transient utility that displays current assignments of CP/M 3 logical devices and the names of the physical devices. DEVICE allows you to assign logical CP/M 3 devices to peripheral devices attached to the computer. The DEVICE command also sets the communications protocol and displays or sets the current console screen size.

CP/M 3 supports the following five logical devices:

CONIN:
CONOUT:
AUXIN:
AUXOUT:
LST:

Display Device Characteristics and Assignments

Syntax: DEVICE {NAMES|VALUES|physical–dev | logical–dev}

Explanation: The preceding form to the DEVICE command displays the names and attributes of the physical devices and the current assignments of the logical devices in the system.

Examples: A > DEVICE

The preceding command displays the physical devices and current assignments of the logical devices in the system. The following is a sample response:

```
A>DEVICE
Physical Devices:
I=Input,O=Output,S=Serial,X=Xon-Xoff
CON108 NONE  IO      SERIAL 300    IOS     PARALL NONE  O
PAR2   NONE  O       SPOOL  NONE   O       USER   NONE

Current Assignments:
CONIN:  = CON108
CONOUT: = CON108
AUXIN:  = SERIAL
AUXOUT: = SERIAL
LST:    = SPOOL

Enter new assignment or hit RETURN
```

A > DEVICE NAMES

The preceding command lists the physical devices with a summary of the device characteristics.

A> DEVICE VALUES

The preceding command displays the current logical device assignments.

A > DEVICE CONSOLE

The preceding command displays the attributes of the physical device CON108.

A > DEVICE LST:

The preceding command displays the assignment of the logical device LST:

Assign a Logical Device

Syntax: DEVICE logical–dev = physical–dev {option} {, physical–dev {option} ,... }
DEVICE logical–dev = NULL

Explanation: The first form assigns a logical device to one or more physical devices. The second form disconnects the logical device from any physical device.

Option	Meaning
XON	refer to the XON/XOFF communications protocol. This protocol uses two special characters in the ASCII character set called XON and XOFF. XON signals transmission on, and XOFF signals transmission off. Before each character is output from the computer to the peripheral device, the computer checks to see if there is any incoming data from the peripheral. If the incoming character is XOFF, the computer suspends all further output until it receives an XON from the device, indicating that the device is again ready to receive more data.
NOXON	indicates no protocol and the computer sends data to the device whether or not the device is ready to receive it.

Examples

```
A>DEVICE CONOUT:=PARALL, CON108
A>DEVICE AUXIN:=SERIAL [XON]
A>DEVICE LST:=NULL
```

The first example assigns the system console output, CONOUT:, to the parallel printer, PARALL, and the screen, CON108. The second example assigns the auxiliary logical input device, AUXIN:, to the physical device SERIAL using protocol XON/OFF. The third example disconnects the list output logical device, LST:

Set Attributes of a Physical Device

Syntax: DEVICE physical–dev {option}

Explanation: The preceding form of the DEVICE command sets the attributes of the physical device specified in the command.

Example: A > DEVICE SERIAL [XON]

The preceding command sets the XON/OFF protocol for the physical device SERIAL.

Display or Set the Current Console Screen Size

Syntax: DEVICE CONSOLE [PAGE COLUMNS=columns LINES=lines]

Explanation: The preceding form of the DEVICE command displays or sets the current console size.

Examples: A > DEVICE CONSOLE [PAGE]
A > DEVICE CONSOLE [COLUMNS=40, LINES=16]

The first example displays the current console page width in columns and length in lines. The second example sets the screen size to 40 columns, and 16 lines.

The DIR Command

Syntax:

DIR {d:}
DIR {filespec}

DIRSYS {d:}
DIRSYS {filespec}

DIR {d:} [options]
DIR {filespec} {filespec} ... [options]

Explanation:

The DIR commands displays the names of files and the attributes associated with the files. DIR and DIRSYS are built–in utilities; DIR with options is a transient utility.

Display Directory

Syntax:

DIR {d:}
DIR {filespec}

DIRSYS {d:}
DIRSYS {filespec}

Explanation:

The DIR and DIRSYS commands display the names of files cataloged in the directory of an on–line disk. The DIR command lists the names of files in the current user number that have the Directory (DIR) attribute. DIR accepts wildcards in the file specification. You can abbreviate the DIRSYS command to DIRS.

The DIRSYS command displays the names of the files in the current user number that have the system (SYS) attribute. Although you can read System (SYS) files that are stored in user 0 from any other user number on the same drive, DIRSYS only displays user 0 files if the current user number is 0. DIRSYS accepts wildcards in the file specification.

If you omit the drive and file specifications, the DIR command displays the names of all files with the DIR attribute on the default drive for the current user number. Similarly, DIRSYS displays all the SYS files.

If the drive specifier is included, but the filename and filetype are omitted, the DIR command displays the names of all DIR files in the current user on the disk in the specified drive. DIRSYS displays the SYS files.

If the file specification contains wildcard characters, all filenames that satisfy the match are displayed on the screen.

If no filenames match the file specification, or if no files are cataloged in the directory of the disk in the named drive, the DIR or DIRSYS command displays the message:

No File

If system (SYS) files match the file specification, DIR displays the message:

SYSTEM FILE(S) EXIST

If nonsystem (DIR) files match the file specification, DIRSYS displays the message:

NON–SYSTEM FILES(S) EXIST

The DIR command pauses after filling the screen. Press any key to continue the display.

Note: You can use the DEVICE command to change the number of columns displayed by DIR or DIRSYS.

Examples:

A > DIR

Displays all DIR files cataloged in user 0 on the default drive A.

A > DIR B:

Displays all DIR files for user 0 on drive B.

A > DIR B:X.BAS

Displays the name X.BAS if the file X.BAS is present on drive B.

4A > DIR *.BAS

Displays all DIR files with filetype BAS for user 4 on drive A.

B > DIR A:X*.C?D

Displays all DIR files for user 0 on drive A whose filename begins with the letter X, and whose three character filetype contains the first character C and last character D.

A DIRSYS

Displays all files for user 0 on drive A that have the system (SYS) attribute.

3A > DIRS *.COM

This abbreviated form of the DIRSYS command displays all SYS files with filetype COM on the default drive A for user 3.

Display Directory with Options

Syntax: DIR {d:} [options]
DIR {filespec} {filespec} ... [option]

Explanation: The DIR command with options is an enhanced version of the DIR command. The DIR command displays CP/M 3 files in a variety of ways. DIR can search for files on any or all drives, for any or all user numbers.

DIR allows the option list to occur anywhere in the command tail. These options modify the entire command line. Only one option list is allowed.

Options must be enclosed in square brackets. The options can be used individually, or strung together separated by commas or spaces. Options can be abbreviated to only one or two letters if the abbreviation unambiguously identifies the option.

If a directory listing exceeds the size of your screen, DIR automatically halts the display when it fills the screen. Press any key to continue the display.

DIR Display Options

Option	Function
ATT	Displays the user-definable file attributes F1, F2, F3, and F4.
DIR	Displays only files that have DIR attribute.
DRIVE=ALL	Displays files on all accessed drives. DISK is also acceptable in place of DRIVE in all the DRIVE options.
DRIVE=(A,....P)	Displays files on the drives.
DRIVE=d	Displays files on the drive specified by d.
EXCLUDE	Displays the files on the default drive and user area that do not match the files specified in the command line.
FF	Sends an initial form-feed to the printer device if the printer has been activated by CTRL-P. If the LENGTH=n option is also specified, DIR issues a form-feed every n line. Otherwise, the FF option deactivates the default paged output display.

FULL	Shows the name of the file and the size of the file. The size is shown as the amount of space in kilobytes and the number of 128–byte records allocated to the file. FULL also shows the attributes of the file. (See the SET command for description of file attributes). If there is a directory label on the drive, DIR shows the password protection mode. The display is alphabetically sorted.
LENGTH=n	Displays n lines of output before inserting a table heading. n must be in the range between 5 and 65536. The default length is one full screen information.
MESSAGE	Displays the names of the specified drives and user numbers it is currently searching. If there are no files in the specified locations, DIR displays the file not found message.
NOPAGE	Continously scrolls information by on the screen. Does not wait for you to press a key to restart the scrolling movement.
NOSORT	Displays files in the order it find them on the disk. If this option is not included, DIR displays the file alphabetically.
RO	Displays only the files that have the Read–Only attribute.

RW	Displays only the files that have the Read–Write attribute.
SIZE	Displays the filename and file size in kilobytes.
SYS	Displays only the files that have the SYS attribute.
USER=n	Displays the file under the user number specified by n.
USER=(0,1...,15)	Displays files under the user numbers specified.

Examples: A > DIR C: [FULL]
A > DIR C: [SIZE]

```
A>DIR [FULL]
Scanning Directory...
Sorting  Directory...
Directory For Drive A:  User  0

    Name     Bytes  Recs  Attributes     Name     Bytes  Recs  Attributes
------------ ------ ------ ------------ ------------ ------ ------ ------------
COPY     COM    2k     11 Dir RW       COPYSYS  COM    2k     12 Dir RW
CPM3     SYS   17k    132 Dir RW       DEVICE   COM    8k     58 Dir RW
DIR      COM   15k    114 Dir RW       DUMP     COM    1k      8 Dir RW
ERASE    COM    4k     29 Dir RW       FORMAT   COM    3k     23 Dir RW
GET      COM    7k     51 Dir RW       PATCH    COM    3k     20 Dir RW
PIP      COM    9k     68 Dir RW       PUT      COM    7k     55 Dir RW
RENAME   COM    3k     23 Dir RW       SAVE     COM    2k     15 Dir RW
SET      COM   11k     81 Dir RW       SETDEF   COM    4k     32 Dir RW
SHOW     COM    9k     66 Dir RW       SUBMIT   COM    6k     42 Dir RW
TYPE     COM    3k     24 Dir RW

Total Bytes     =    116k  Total Records =     864  Files Found =   19
Total 1k Blocks =    116   Used/Max Dir Entries For Drive A:   20/  64
```

```
A>DIR [SIZE]
Scanning Directory...
Sorting  Directory...
Directory For Drive A:  User  0
A: COPY     COM    2k : COPYSYS  COM    2k : CPM3     SYS   17k
A: DEVICE   COM    8k : DIR      COM   15k : DUMP     COM    1k
A: ERASE    COM    4k : FORMAT   COM    3k : GET      COM    7k
A: PATCH    COM    3k : PIP      COM    9k : PUT      COM    7k
A: RENAME   COM    3k : SAVE     COM    2k : SET      COM   11k
A: SETDEF   COM    4k : SHOW     COM    9k : SUBMIT   COM    6k
A: TYPE     COM    3k
Total Bytes     =    116k  Total Records =      864  Files Found =   19
Total 1k Blocks =    116   Used/Max Dir Entries For Drive A:   20/  64
```

Both the full format and the size format follow their display with two lines of totals. The first line displays the total number of kilobytes, the total number of records, and the total number of files for that drive and user area. The second line displays the total number of 1k blocks needed to store the listed files. The number of 1k blocks shows the amount of storage needed to store the files on a single density disk, or on any drive that has a block size of one kilobyte. The second line also shows the number of directory entries used per number of directory entries available on the drive.

A > DIR [DRIVE=C, FF]

DIR sends a form–feed to the printer before displaying the files on drive C.

A > DIR D: [RW, SYS]

The preceding example displays all the files on drive D with Read–write and SYS attributes.

A > DIR C: [USER=ALL]

Displays all the files under each user number (0–15) on drive C.

A > DIR [USER 2]

Displays all the files under user 2 on the default drive.

A > DIR C: [USER=(3,4,10)]

This example displays all the files under user number 3, 4, and 10 on drive C.

A > DIR [DRIVE=ALL]

Displays all the files under user 0 on all the drives in the drive search again. (See the SETDEF command.)

4A > DIR [DRIVE=C]

Displays all the files under user 4 on drive C.

A > DIR [DRIVE=(B,D)]

Displays all the files under user 0 and drive B and D.

A > DIR [EXCLUDE] *.COM

The preceding example above lists all the files on the default drive and user 0 that do not have a filetype of COM.

A>DIR [user=all,drive=all,sys]*.PLI*.
COM *.ASM

The preceding command line instructs DIR to list all the system files of type PLI, COM, and ASM on the system in the currently active drives for all the user numbers on the drives.

A > DIR X.SUB [MESSAGE, USER=ALL,
DRIVE ALL]

The preceding example instructs DIR to display the filename TESTFILE.BOB if it is found on any logged– in drive for any user number.

A > DIR [drive=all user=all] TESTFILE.BOB

The preceding command searches all drives under each user number for X.SUB. During the search, DIR displays the drives and user numbers.

A > DIR [size,rw] D:

The preceding example instructs DIR to list each Read–Write file that resides on drive D with its size in kilobytes. Note that D: is equivalent to D:*.*.

The DUMP Command

Syntax: DUMP filespec

Explanation: Dump displays the contents of a file in hexadecimal and ASCII format.

Example: A > DUMP ABC.TEX

Console output can look like the following:

```
DUMP – Version 3.0
0000: 41 42 43 0D 0A 44 45 46 0D 0A 47 48 49 0D 0A 1A ABC.....
......
......
```

The ERASE Command

Syntax: ERASE {filespec} {[CONFIRM]}

Explanation: The ERASE command removes one or more files from a disk's directory in the current user number. Wildcard characters are accepted in the filespec. Directory and data space are automatically reclaimed for later use by another file. The ERASE command can be abbreviated to ERA.

Use the ERASE command with care because all files in the current user number that satisfy the file specification are removed from the disk directory.

Command lines that take the form

ERASE {d:} wildcard–filespec

require your confirmation because they erase an entire group of files, not just one file. The system prompts with the following message:

ERASE {d:} wildcard–filespec (Y/N)?

Respond with y if you want to remove all matching files, and n if you want to avoid erasing any files.

If no files match the file specification, you see the following message:

No file

The CONFIRM option informs the system to prompt for verification before erasing each file that matches the filespec. You can abbreviate CONFIRM to C.

If you use the CONFIRM option with wildcard–filespec, then ERASE prompts for confirmation for each file. You can selectively erase the files you want by responding Y to the confirm message, or keep the files by responding N to the confirm message.

Example: A > ERASE X.PAS

This command removes the file X.PAS from the disk in drive A.

A > ERA *.PRN

The system asks to confirm:

ERASE *.PRN (Y/N)?Y

All files with the filetype PRN are removed from the disk in drive A.

B > ERA A:MY*.* [CONFIRM]

Each file on drive A with a filename that begins with MY is displayed with a question mark for confirmation. Type Y to erase the file displayed, N to keep the file.

A > ERA B:*.*
ERASE B:*.* (Y/N)?Y

All files on drive B are removed from the disk.

The FORMAT Command

Syntax:	FORMAT
Explanation:	The FORMAT command allows you to get a diskette ready to receive information. This command checks the diskette for bad spots, built a directory to hold information about the files.
Example:	A > FORMAT

FORMAT Vers 1.0

40–Tracks–Drive Formatter

Format which disk?B

Format 40 Tracks?Y

Press < RETURN > to begin

Formatted

Format another disk?N

The preceding example formats the diskette in drive B. As you know, a disk has 40 tracks, it is up to individual to format number of tracks. To format another disk, press Y for the last message displayed.

The GET Command

Syntax: GET {CONSOLE INPUT FROM} FILE filespec {[ECHO | NO ECHO} | SYSTEM]}

GET {CONSOLE INPUT FROM} CONSOLE

Explanation: The GET command is a transient utility that directs CP/M 3 to take console input from a file. The file can contain CP/M 3 system commands and/or input for a user program. If you use the SYSTEM option, GET immediately takes the next system command from the file.

Console input is taken from a file until the program terminates. If the file is exhausted before the program is terminated, the program looks for subsequent input from the console. If the program terminates before exhausting all its input, the system reverts back to the console for console input.

When the SYSTEM option is used, the system immediately goes to the file specified for console input. If you omit the SYSTEM option, you can enter one system command to initiate a user program whose console input is taken from the file specified in the GET command. The system reverts to the console for input when it reaches the end of the GET file input. The system also reverts to the console for console input if a GET CONSOLE INPUT FROM CONSOLE command is included in the input file.

Get Console Input from a File

Syntax: GET { CONSOLE INPUT FROM } FILE filespec {[options]}

Explanation: The preceding form of the GET command tells the system to get subsequent console input from a file. The following table lists the GET options that you use in the following format:

[{ECHO NO ECHO} SYSTEM]

Get options

Option	Meaning
ECHO	Specifies that the input is echoed to the console. This is the default option.
NO ECHO	Specifies that the file input is not to be echoed to the console. The program output and the system prompts are not affected by this option and are still echoed to the console.
SYSTEM	Specifies that all system input is to be taken from the disk file specified in the command line. GET takes system and program input from the file until the file is exhausted or until GET reads a GET console command from the file.

Examples:

A > GET FILE XINPUT
A > MYPROG

The preceding sequence of commands tells the system to activate the GET utility. However, because SYSTEM is not specified, the system reads the next input line from the console and executes MYPROG. If MYPROG program requires console input, it is taken from the file XINPUT. When MYPROG terminates, the system reverts to the console for console input.

A > GET FILE XIN2 [SYSTEM]

The preceding command immediately directs the system to get subsequent console input from file XIN2 because it includes the SYSTEM option. The system reverts to the console for console input when it reaches the end of the file in XIN2. Or, XIN2 can redirect the system back to the console if it contains a GET CONSOLE command.

Terminate Console Input from a File

Syntax: GET {CONSOLE INPUT FROM} CONSOLE

Explanation: The preceding form of the GET command tells the system to get console input from console.

Example: A > GET CONSOLE

The preceding GET command tells the system to get console input from the console. You can use this command in a file (previously specified in a GET FILE command) which is already being read by the system for console input. It is used to redirect the console input to the console before the end of the file is reached.

The PATCH Command

Syntax: PATCH filename { typ } { n }

Explanation: The PATCH command displays or installs patch number n to the CP/M 3 system or CP/M 3 command files.

Only CP/M 3 system files of filetype COM, PRL, or SPR can be patched with the PATCH command. If the typ option is not specified, the PATCH utility looks for a file with filetype of COM.

The patch number n must be between 1 and 32 inclusive.

Examples: A > PATCH SHOW 2

The preceding command patches the system SHOW.COM file with patch number 2. The system displays the following question:

Do you want to indicate that Patch #2
has been installed for SHOW.COM?Y

If the patch is successful, the system displays the message:

Patch Installed

If the patch is not successful, the system displays the following message:

Patch not Installed

One of the following error messages might be displayed:

ERROR: Patch requires CP/M 3
ERROR: Invalid filetype typ.
ERROR: Serial Number mismatch
ERROR: Invalid patch number n.

The PIP Command

Syntax: PIP dest–filespec | d: {[Gn]}=src–filespec {[o]} {,...} | d: {[o]}

Explanation: PIP (Peripheral Interchange Program) is a transient utility that copies one or more files from one disk and or user number to another. PIP can rename a file after copying it. PIP can combine two or more files into one file. PIP can also copy a character file from disk to the printer or other auxiliary logical output device. PIP can create a file on disk from input from the console or other logical input device. PIP can transfer data from a logical input device to a logical output device.

PIP copies file attributes with the file. This includes Read–Write or Read–Only and SYS or DIR file attributes and the user–definable attributes F1 through F4. If a file is password–protected, you must enter the password in the command line following the filename and/or filetype to which it belongs. If the password fails, the file is skipped and the failure noted.

When you specify a destination file with a password, PIP assigns that password to the destination file and automatically sets the password protection mode to READ. When you specify a destination file with no password, PIP does not assign a password to the destination file. When you specify only a destination drive, PIP assigns the same password and password protection mode to the destination file as specified in the source file. When you specify a destination file with a password, PIP automatically sets the password protection mode to READ. This means that you need a password to read the file. (See the SET command.)

Single File Copy

Syntax:

PIP d: {[Gn]} = src–filespec {[options]}
PIP dest–filespec {[Gn]} = d: {[options]}
PIP dest–filespec {[Gn]} = src–filespec {[o]}

Explanation:

The first form shows the simplest way to copy a file. PIP looks for the file named by src–filespec on the default or optionally specified drive. PIP copies the file to the drive specified by d: and gives it the name specified by src–filespec. If you want, you can use the [GN] option to place your destination file (dest–filespec in the user number specified by n. The only option recognized for the destination file is [GN]. Several options can be combined together for the source file specification (src–filespec).

The second form is variation of the first. PIP looks for the file named by dest–filespec on the drive specified by d:, copies it to the default or optionally specified drive, and gives it the name specified by dest–filespec.

Remember that PIP always goes to and gets from the current default user number unless you specify otherwise with the [GN] option.

Before you start PIP, be sure that you have enough free space in kilobytes on your destination disk to hold the entire file or files that you are copying. Even if you are replacing an old copy on the destination disk with a new copy, PIP still needs enough room for the new copy before it deletes the old copy. Use the DIR command to determine filesize and the SHOW command to determine disk space. If there is not enough space, you can delete the old copy first by using the ERASE command.

Data is first copied to a temporary file to ensure that the entire data file can be constructed in the space available on the disk. PIP gives the temporary file and filename specified for the destination, with the filetype $$$. If the copy operation is successful, PIP changes the temporary filetype $$$ to the filetype specified in the destination.

If the copy operation succeeds and a file with the same name as the destination file already exists, the old file with the same name is erased before renaming the temporary file.

File attributes (DIR, SYS, RO, RW) are transferred with the files.

If the existing destination file is set to Read–Only (RO), PIP asks you if you want to delete it. Answer Y or N. Use the [W] option to write over Read–Only files.
You can include PIP options following each source name. There is one valid option ([GN]-- go to user number n) for the destination file specification. Options are enclosed in square brackets. Several options can be included for source files. They can be packed together or separated by spaces. Options can verify that a file was copied correctly, allow PIP to put read a file with the system (SYS) attribute, cause PIP to put a file into or copy it from a specified user number, transfer from lower–to upper–case, and much more.

Examples:

A > PIP B:=A:oldfile.dat
A > PIP B:oldfile.dat=A:

Both forms of this command cause PIP to read the file oldfile.dat from drive A and put an exact copy of it onto drive B. This is called short form of PIP, because the source or destination names only a drive and does not include a filename. When using this form you cannot copy a file from one drive and user number to the same drive and user number. You must put the destination file on a different drive or in a different user number. (See the section on PIP Options, and the USER command.) The second short form produces exactly the same result as the first one.

A > PIP B:newfile.dat=A:oldfile.dat

This command copies the file oldfile.dat from drive A to drive B and renames it to newfile.dat. The file remains as oldfile.dat on drive A. This is the long form of the PIP command, because it names a file on both sides of the command line.

A > PIP newfile.dat = oldfile.dat

Using this long form of PIP, you can copy a file from one drive and user number (usually user 0 because CP/M 3 automatically starts out in user 0 – the default user number) to the same drive and user number.

A > PIP B:PROGRAM.BAK = A:PROGRAM.DAT [G1]

The preceding command copies the file PROGRAM.DAT from user 1 on drive A to the current selected user number on drive B and renames the filetype on drive B to BAK.

B > PIP program2.dat = A:program1.dat [E V G3]

In this comamnd, PIP copies the file named program1.dat on drive A and echoes [E] the transfer to the console, verifies [V] that the two copies are exactly the same, and gets [G3] the file program 1. dat from user 3 on drive A. Because there is no drive specified for the destination, PIP automatically copies file to the default user number and drive, in this case user 0 and drive B.

Multiple File Copy

Syntax: PIP d: {[Gn]} = { d: } wildcard–filespec {[options]}

Explanation: When you use a wildcard in the source specification, PIP copies matching files one–by–one to the destination drive, retaining the original name of each file. PIP displays the message COPYING followed by each filename as the copy operation proceeds. PIP issues an error message and aborts the copy operation if the destination drive and user number are the same as those specified in the source.

Examples: A > PIP B:=A:*.COM

This command causes PIP to copy all the files on drive A with the filetype COM to drive B.

A > PIP B:=A:*.*

This command causes PIP to copy all the files on drive A to drive B. You can use this command to make a back–up copy of your distribution disk. Note, however, that this command does not copy the CP/M 3 system from the system tracks. COPYSYS copies the system for you.

A > PIP B:=A:PROG????.*

The preceding command copies all files whose filenames begin with PROG from drive A to drive B.

A > PIP B:[G1]=A:*.BAS

This command causes PIP to copy all the files with a filetype of BAS on drive A in the default user number (user 0) to drive B in user number 1.

Combining Files

Syntax: PIP dest–filespec {[Gn]} = src–filespec {[o]}, src–filespec {[o]} {,...}

Explanation: This form of the PIP command lets you specify two or more files in the source. PIP copies the files specified in the source from left to right and combines them into one file with the name indicated by the destination file specification. This procedure is called file concatenation. You can use the [GN] option after destination file to place it in the user number specified by n. You can specify one or more options for each source file.

Some of the options force PIP to copy files character–by–character. In these cases, PIP looks for a CTRL–Z character to determine where the end of the file is. All of the PIP options force a character transfer except the following:

A, C, Gn, K, O, R, V, and W.

Copying data to or from logical devices also forces a character transfer.

You can terminate PIP operations by typing CTRL–C.

After concatenating files, PIP only searches the last record of a file for the CTRL–Z end–of–file character. However, if PIP is doing a character transfer, it stops when it encounters a CTRL–Z character.

Use the [O] option if you are concatenating machine code files. The [O] option causes PIP to ignore embedded CTRL–Z (end–of–file) character, comma which indicate the end–of–file character in text files, but might be valid data in object code files.

Examples: A > PIP NEWFILE=FILE1, FILE2, FILE3

The three files named FILE1, FILE2 and FILE3 are joined from left to right and copied to NEWFILE.$$$. NEWFILE.$$$ is renamed to NEWFILE upon successful completion of the copy operation. All source and destination files are on the disk in the default drive A.

A > PIP B:X.BAS = Y.BAS, B:Z.BAS

The file Y.BAS on drive A is joined with Z.BAS from drive B and placed in the temporary file X.$$$ on drive B. The file X.$$$ is renamed to X.BAS on drive B when PIP runs to successful completion.

Copy Files to and from Auxiliary Devices

Syntax: PIP dest–filespec {[Gn] } = src–filespec {[o]}

CONOUT:	CONIN:
AUXOUT:	AUXIN:
LST:	

Explanation: This form is a special case of the PIP command line that lets you copy a file from a disk to a device, from a device to a disk or from one device to another. The files must contain printable characters. Each peripheral device is assigned to a logical device that identifies a source device that can transmit data or a destination device that can receive data. (See the DEVICE command.) A colon follows each logical device name so it cannot be confused with a filename. Enter CTRL–C to abort a copy operation that uses a logical device in the source or destination.

The logical device names are listed as follows:

CON: Console input or output device. When used as a source, usually the keyboard; when used as a destination, usually the screen.

AUX: Auxiliary Input or Output Device.

LST: The destination device assigned to the list output device, usually the printer.

Examples: A > PIP B:FUNFILE.SUE = CON:

Whatever you type at the console keyboard is written to the FUNFILE.SUE on drive b. End the keyboard input by typing a CTRL–Z.

Note that when the CON: device is the source you must enter both the carriage return (RETRN) and line–feed (LINE FEED) keys for a new line.

A > PIP LST:=CON:

Whatever you type at the console keyboard is written to the list device, generally the printer. Terminate input with a CTRL–Z.

B > PIP PRN:=CON:,MYDATA.DAT

Characters are first read from the console input device, generally the keyboard, and sent directly to your printer device. You type a CTRL–Z character to tell PIP that keyboard input is complete. At that time, PIP continues by reading character data from the file MYDATA .DAT on drive B.

A > PIP LST:=B:DRAFT.TXT[T8]

The file DRAFT.TXT on drive B is written to the printer device. Any tab characters are expanded to the nearest column that is a multiple of 8.

Syntax: PIP

Explanation: This form of the PIP command starts the PIP utility and lets you type multiple command lines while PIP remains in user memory.

PIP writes an asterisk on your screen when ready to accept input command lines.

You can type any valid command line described under previous PIP formats following the asterisk prompt.

Terminate PIP by pressing only the RETRN key following the asterisk prompt.

Examples:

```
A > PIP
CP/M 3 PIP VERSION 3.0
*NEWFILE=FILE1,FILE2,FILE3
*APROG.COM=BPROG.COM
*A:=B:X.BAS
*B:=*.*
* < RETRN >
A >
```

This command loads the PIP program. The PIP command input prompt, *, tells you that PIP is ready to accept commands. The effects of this sequence of commands are the same as in the previous example, where the command line is included in the command tail. PIP is not loaded into memory for each command. To exit this PIP command mode, press RETRN or one of its equivalent control characters, CTRL–J or CTRL–M.

Using Options with PIP

Explanation: With options you can process your source file in special ways. You can expand tab characters, translate from upper– to lower–case, extract portions of your text, verify that the copy is correct, and much more.

The PIP options are listed in the following table using n to represent a number and s to represent a sequence of characters terminated by a CTRL–Z. An option must immediately follow the file or device it affects.

The option must be enclosed in square brackets []. For those options that require a numeric value, no blanks can occur between the letter and the value.

You can include the [GN] option after a destination file specification. You can include a list of options after a source file or source device. An option list is a sequence of single letters and numberic values that are optionally separated by blanks and enclosed in square brackets [].

PIP Options

Option	Function
A	Copy only the files that have been modified since the last copy. To back up only the files that have been modified since last back–up, use PIP with archive option, [A].
C	Prompt for confirmation before performing each copy operation. Use the [C] option when you want to copy only some files of a particular filetype.
Dn	Delete any characters past column n. This parameter follows a source file that contains lines too long to be handled by the destination device, for example, an 80–character printer or narrow console. The number n should be the maximum column width of the destination device.
E	Echo transfer at console. When this parameter follows a source name, PIP displays the source data at the console as the copy is taking place. The source must contain character data.
F	Filter form–feeds. When this parameter follows a source name, PIP removes all form–feeds embedded in the source data. To change form–feeds set for one page length in the source file to another page length in the destination file, use the F command to delete the old form–feeds and a P command to simultaneously add new form–feeds to the destination file.

Gn	Get source from or go to user number n. When this parameter follows a source name, PIP searches the directory of user number n for the source file. When it follows the destination name, PIP places the destination file in the user number specified by n. The number must be in the range 0 to 15.
H	Hex data transfer. PIP checks all data for proper Intel hexadecimal file format. The console displays error messages when errors occur.
I	Ignore :00 records in the transfer of Intel hexadecimal format file. The I option automatically sets the H option.
L	Translate upper–case alphabetics in the source file to lower–case in the destination file. This parameter follows the source device or filename.
N	Add line numbers to the destination file. When this parameter follows the source filename, PIP adds a line number to each line copied, starting with 1 and incrementing by one. A colon follows the line number. If N2 is specified, PIP adds leading zeros to the line number and inserts a tab after the number. If the T parameter is also set, PIP expands the tab.
O	Object file transfer for machine code (noncharacter and therefore nonprintable) files. PIP ignores any CTRL–Z end–of–file during concatenation and transfer. Use this option if you are combining object code files.

Pn	Set page length. n specifies the number of lines per page. When this parameter modifies a source file, PIP includes a page eject at the beginning of the destination file and at every n lines. If n=1 or is not specified, PIP inserts page ejects every sixty lines.
Qs	Quit copying from the source device after the string s. When used with the S parameter, this parameter can extract a portion of a source file. The string argument must be terminated by CTRL–Z.
R	Read system (SYS) files. Usually, PIP ignores files marked with the system attribute in the disk directory. But when this parameter follows a source filename, PIP copies system files, including their attributes, to the destination.
Ss	Start copying from the source device at the string s. The string argument must be terminated by CTRL–Z. When used with the Q parameter, this parameter can extract a portion of a source file. Both start and quit strings are included in the destination file.
Tn	Expand tabs. When this parameter follows a source filename, PIP expands tab (CTRL–I) characters in the destination file. PIP replaces each CTRL–I with enough spaces to position the next character in a column divisible by n.
U	Translate lower–case alphabetic characters in the source file to upper–case in the destination file. This parameter follows the source device or filename.

V	Verify that data has been copied correctly. PIP compares the destination to the source data to ensure that the data has been written correctly. The destination must be a disk file.
W	Write over files with RO (Read-Only) attribute. Usually, if a PIP command tail includes an existing RO file as a destination, PIP sends a query to the console to make sure you want to write over the existing file. When this parameter follows a source name, PIP overwrites the RO file without a console exchange. If the command tail contains multiple source files, this parameter need follow only the last file in the list.
Z	Zero the parity bit. When this parameter follows a source name, PIP sets the parity bit of each data byte in the destination file to zero. The source must contain character data.

Examples:

A > NEWPROG. BAS=CODE. BAS[L], DATA. BAS[U]

This command contructs the file NEWPROG-.BAS on drive A by joining the two file CODE.BAS and DATA.BAS from drive A. During the copy operation, CODE.BAS is translated to lower–case, while DATA.BAS is translated to upper–case.

A > PIP CON:=WIDEFILE.BAS[D80]

This command writes the character file WIDE-FILE.BAS from drive A to the console device, but deletes all characters following the 80th column position.

A > PIP B:=LETTER.TXT[E]

The file LETTER. TXT from drive A is copied to LETTER. TXT on drive B. The LETTER. TXT file is also written to the screen as the copy operation proceeds.

A > PIP LST:=A:LONGPAGE.TXT[FP65]

This command writes the file LONG-PAGE.TXT from drive B to the printer device. As the file is writtern, form–feed characters are removed and reinserted at the beginning and every 65th line thereafter.

B > PIP LST:=PROGRAM.BAS[NT8U]

This command writes the file PROGRAM. BAS from drive B to the printer device. The N parameter tells PIP to number each line. The T8 parameter expands tabs to every eighth column. The U parameter translates lower–case letters to upper–case as the file is printed.

A > PIP PORTION. TXT=LETTER.TXT
[SDear Sir ^ Z QSincerely ^ Z]

This command abstracts a portion of the LETTER. TXT file from drive A by searching for the character sequence “Dear Sir” before starting the copy operation. When found, the characters are copied to PORTION. TXT on drive A until the sequence “Sincerely” is found in the source file.

B > PIP B:=A:*.COM[VWR]

This command copies all files with filetype COM from drive A to drive B. The V parameter tells PIP to read the destination files to ensure that data was correctly transferred. The W parameter lets PIP overwrite any destination files that are marked as RO (Read–Only). The R parameter tells PIP to read files from drive A that are marked with the SYS (System) attribute.

The PUT Command

Syntax:

PUT CONSOLE { OUTPUT TO } FILE filespec { o }
PUT PRINTER { OUTPUT TO } FILE filespec { o }
PUT CONSOLE { OUTPUT TO } CONSOLE
PUT PRINTER { OUTPUT TO } PRINTER

Explanation:

The PUT command is a transient utility that lets you direct console output or printer output to a file. PUT allows you to direct the system to put console output or printer output to a file for the next system command or user program entered at the console. Or, PUT directs all subsequent console or printer output to a file when you include the SYSTEM option.

Console output is directed to a file until the program terminates. Then, console output reverts to the console. Printer output is directed to a file until the program terminates. Then printer output is directed back to the printer.

When you use the SYSTEM option, all subsequent console/printer output is directed to the specified file. This option terminates when you enter the PUT CONSOLE or PUT PRINTER command.

The syntax for the option list is

[{ECHO | NO ECHO} {FILTER | NO FILTER} {SYSTEM}]

The following table defines the preceding option list.

PUT Options

Option	Meaning
ECHO	Specifies that the output is echoed to the console. ECHO is the default option when you direct console output to a file.
NO ECHO	Specifies that the file output is not to be echoed to the console.

FILTER	Specifies that filtering of control characters is allowed, which means that control characters are translated to printable characters. For example, an escape character is translated to ^ [.
NO FILTER	Means that PUT does not translate control characters. This is the default option.
SYSTEM	Specifies that system output and program output is written to the file specified by filespec. Output is written to the file until a subsequent PUT CONSOLE command redirects console output back to the console.

Direct Console Output to a FILE

Syntax: PUT CONSOLE {OUTPUT} TO FILE filespec {[o]}

Explanation: The preceding form of the PUT command tells the system to direct subsequent console output to a file.

Example: A > PUT CONSOLE OUTPUT TO FILE XOUT [ECHO]

The preceding command directs console output to file XOUT with the output echoed to the console.

Put Printer Output to a File

Syntax: PUT PRINTER {OUTPUT TO} FILE filespec {o}

Explanation: The preceding form of the PUT command directs printer output to a file.

The options are the same as in the PUT CONSOLE command, except that option NO ECHO is the default for the PUT PRINTER command. Note that if ECHO is specified, printer output is echoed to the printer.

Examples: A > PUT PRINTER OUTPUT TO FILE XOUT

A > MYPROG

The preceding example directs the printer output of program MYPROG to file XOUT. The output is not echoed to the printer.

A > PUT PRINTER OUTPUT TO FILE XOUT2 [ECHO, SYSTEM]

The preceding command directs all printer output to file XOUT2 and to the printer, and the PUT is in effect until you enter a PUT PRINTER OUTPUT TO PRINTER command.

The printer output can be directed to one or more files. The output to these file is terminated when you revert printer output to the printer using the following command:

PUT PRINTER OUTPUT TO PRINTER

Terminate Console Output to a File

Syntax: PUT CONSOLE {OUTPUT TO} CONSOLE

Explanation: The preceding form of the PUT command directs console output to the console.

Terminate Printer Output to a File

Syntax: PUT PRINTER {OUTPUT TO} PRINTER

Explanation: The preceding form of the PUT command directs the printer output to printer.

Example: A > PUT PRINTER OUTPUT TO PRINTER

The preceding example directs printer output to the printer.

The RENAME Command

Syntax: RENAME {new–filespec=old–filespec}

Explanation: The RENAME command lets you change the name of a file that is cataloged in the directory of a disk. It also lets you change several filenames if you use wildcards in the filespecs. You can abbreviate RENAME to REN.

The new–filespec must not be the name of any existing file on the disk. The old–filespec identifies an existing file or files on the disk.

The RENAME command changes the file named by old–filespec to the name given as new–filespec.

RENAME does not make a copy of the file. RENAME changes only the name of the file.

If you omit the drive specifier, RENAME assumes the file to rename is on the default drive. You can include a drive specifier as a part of the newname.
If both file specifications name a drive, it must be the same drive.

If the file given by oldname do not exist, RENAME displays the following message on the screen: No File

If the file given by newname is already present in the directory, RENAME displays the following message on the screen:

ERROR: Not renamed: {filename.typ} file already exists, delete (Y/N)?

If you want to delete the old file, type Y to delete. Otherwise, type N to keep the old file and not rename the new file.

If you use wildcards in the filespecs, the wildcards in the filespec must correspond exactly to the wildcards in the old filespec. For example, in the following two commands, the wildcard filespecs correspond exactly:

```
A > REN *.TX1=*.TEX
A > REN A*.T*=S*.T*
```

In the following example, the wildcards do not match and CP/M 3 returns an error message.

A > REN A*.TEX=A.T*

Examples:

A > RENAME NEWASM.BAS=
OLDFILE.BAS

The file OLDFILE.BAS changes the NEWASM.BAS on drive A.

A > RENAME

The system prompts for the filespecs:

Enter New Name:X.PRN
Enter Old Name:Y.PRN

A >

File Y.PRN is renamed X.PRN on drive A.

A > REN A:X.PAS = Y.PLI

The file Y.PLI changes to X.PAS on drive A.

A > RENAME S*.TEX=A*.TEX

The preceding command renames all the file matching the wildcard A*.TEX to files with filenames matching the wildcard S.TEX, respectively.

A > REN B:NEWLIST=B:OLDLIST

The file OLDLIST changes to NEWLIST on drive B. Because the second drive specifer, B: is implied by the first, it is unnecessary in this example. The preceding command line has the same effect as the following:

A > REN B:NEWLIST=OLDLIST
or
A > REN NEWLIST=B:OLDLIST

The SAVE Command

Syntax: SAVE

Explanation: The SAVE command copies the contents of memory to a file. To use the SAVE utility, first issue the SAVE command, then run your program which reads a file into memory. When your program exits, it exits to the SAVE utility. The SAVE utility prompts you for the filespec to which the memory is to be copied, and the beginning and ending address of the memory to be saved.

Example:

```
A > SAVE
```

The preceding command activates the SAVE utility. Now enter the name of the program that loads a file into memory.

```
A > DDT dump.com
```

Next, execute the program

```
-g0
```

When the program exits, SAVE intercepts the return to the system and prompts you for the filespec and the bounds of memory to be saved.

```
SAVE Ver 3.0
File (or RETURN to exit)?dump2.com
Delete dump2.com? Y
From? 100
To? 400

A >
```

The contents of memory from 100H, hexadecimal, to 400H is copied to file DUMP2.COM.

The SET Command

Syntax:	SET [options] SET d: [options] SET filespec [options]
Explanation:	The SET command initiates password protection of file in the CP/M 3 system. It also sets file and drive attributes, such as the Read–Only, SYS, and user–definable attributes. It lets you label a disk and password protect the label. The SET command include options that affect the disk directory, the drive, or a file or set of files. The discussion of the SET command explicitly states which of the three categories are affected.

Set File Attributes

Syntax:	SET filespec [attribute–options]
Explanation:	The preceding SET command sets the specified attributes of a file or a group of files.

SET File Attributes

Option	Meaning
DIR	Sets the file from the SYS directory to the (DIR) attribute.
SYS	Gives the file the System SYS attribute
RO	Sets the file attribute to allow Read–Only access.
RW	Sets the file attribute to allow Read–Write access.

ARCHIVE=OFF	Sets the archive attribute to off. This means that the file has not been backed up(archived). PIP with the [A] option can copy files with the archive attribute set to OFF. PIP with this option requires an ambiguous filespec and copies only files that have been created or changed since the last time they were backup up with the PIP [A] option. PIP then sets the archive attribute to ON for each line successfully copied.
ARCHIVE=ON	Sets the archive attribute to on. This means that the file has been backed up (archived). The archive attribute can be turned on explicitly by the SET command, or it can be turned on by PIP when copying a group of files with the PIP [A] option. The archive attribute is displayed by DIR.
F1=ON/OFF	Turns on or off the user–definable file attribute F1.
F2=ON/OFF	Turns on or off the user–definable file attribute F2.
F3=ON/OFF	Turns on or off the user–definable file attribute F3.
F4=ON/OFF	Turns on or off the user–definable file attribute F4.

Example: A > SET MYFILE. TEX [RO SYS]

The preceding command sets MYFILE.TEX to Read–Only and System.

A > SET MYFILE.TEX [RW DIR]

The preceding command sets MYFILE.TEX to Read–Write with the Directory (DIR) attribute.

Set Drive Attribute

Syntax: SET {d:} [RO]
SET {d:} [RW]

Explanation: The preceding SET commands set the specified drive to Read–Only or Read–Write.

If a drive is set to Read–Only, PIP cannot copy a file to it, ERASE cannot delete a file from it, RENAME cannot rename a file on it. You cannot perfrom any operation that requires writing to the disk. When the specified drive is set to Read–Write, you can read or write to the disk in that drive. If you enter a CTRL–C to the system prompt, all drives are reset to Read–Write.

Example: A > SET B: [RO]

The preceding command sets drive B to Read–Only.

Assign a Label to the Disk

Syntax: SET {d:} [NAME=labelname.typ]

Explanation: The preceding SET command assigns a label (name) to the disk in the specified or default drive.

CP/M 3 provides a facility for creating a directory label for each disk. The directory label can be assigned an eight–character name and a three–character type similar to a filename and filetype. Label names make it easier to catalog disks and keep track of different disk directories. The default label name is LABEL.

Example: A > SET [NAME=DISK100]

The preceding example labels the disk on the default drive DISK100.

Assign Password to the Label

Syntax:

SET [PASSWORD=password]
SET [PASSWORD=<cr>

Explanation: The first form of the preceding SET command assigns a password to the disk label. The second form of the command removes password protection from the label.

You can assign a password to the label. If the label has no password, any user who has access to the SET program can set other attributes to the disk which might make the disk inaccessible to you. However, if you assign a password to the label, then you must supply the password to set any of the functions controlled by the label. SET always prompts for the password if the label is password–protected.

Examples:

A > SET [PASSWORD=SECRET]
A > SET [PASSWORD=<cr>

The first command assigns SECRET to the disk label. The second command nullifies the existing password.

Note: If you use password protection on your disk, be sure to record the password. If you forget the password, you lose access to your disk or files.

Enable/Disable Password Protection for Files on a Disk

Syntax:

SET [PROTECT=ON]
SET [PROTECT=OFF]

Explanation: The first form of the SET command turns on password protection for all the files on the disk. The password protection must be turned on before you can assign passwords to individual files or commands.

The second SET command disables password protection for the files on your disk.

After a password is assigned to the label and the PROTECT option is turned on, you are ready to assign passwords to your files.

You can always determine if a disk is password–protected by using the SHOW command to display the label.

Assign Passwords to Files

Syntax: SET filespec [PASSWORD=password]

Explanation: The preceding SET command sets the password for filespec to the password indicated in the command tail. Passwords can be up to eight characters long. Lower–case letter are translated to upper–case.

You can use wildcards in the filespec. SET assigns the specified password to the files that match the wildcard–filespec.

Note: Always record the passwords that you assign to your files. Without the password, you cannot access those files unless password protection is turned off for the whole disk. If you forget the password to the directory label, you cannot turn off the password protection for the disk.

Example:

```
A > SET MYFILE.TEX
[PASSWORD=MYFILE]
```

MYFILE is the password assigned to the file MYFILE.TEX.

Set Password Protection Mode for File with Passwords

Syntax:	SET filespec [PROTECT=READ] SET filespec [PROTECT=WRITE] SET filespec [PROTECT=DELETE] SET filespec [PROTECT=NONE]
Explanation:	You can assign one of four modes of password protection to your file. The protection modes are READ, WRITE, DELETE, and NONE and are described in the following table.

Password Protection Modes

Mode	Protection
READ	The password is required for reading, copying, writing, deleting, or renaming the file.
WRITE	The password is required for writing, deleting, or renaming the file. You do not need a password to read the file.
DELETE	The password is only required for deleting or renaming the file. You do not need a password to read or modify the file.
NONE	No password exists for the file. If a password exists, this modifier can be used to delete the password.

Example: B > SET *.TEX [PASSWORD=SECRET, PROTECT=WRITE]

The preceding command assigns the password SECRET to all the TEX files on drive B: Each TEX file is given a WRITE protect mode to prevent unauthorized editing.

Assign a Default Password

Syntax: SET [DEFAULT=password]

Explanation: The preceding set command assigns a default password for the system to use during your computer session. The system uses the default password to access password–protected files if you do not specify a password, or if you enter an incorrect password. The system lets you access the file if the default password matches the password assigned to the file.

Example: A > SET [DEFAULT=dd]

The preceding command instructs the system to use dd as a password if you do not enter a password for a password–protected file.

Additional SET Examples

Examples: A > SET *.COM [SYS, RO, PASS=123, PROT=READ]

The preceding setting gives the most protection for all the COM files on drive A. With the password protection mode set to READ, you cannot even read one of the COM files without entering the password 123, unless the default password has been set to 123. Even if the correct password is entered, you still cannot write to the file because the file is read only.

A > SET *.COM [RW, PROTECT=ONE, DIR]

The preceding command reverses the protection and access attributes of the COM files affected by the previous example. After executing the preceding command, there is no password protection, the files of type COM can be read from or written to, and are set to DIR files.

The SETDEF Command

Syntax: SETDEF {d: {,d: {,d: {,d: }}}}
{[TEMPROARY=d:] [ORDER=(typ {,typ})]}

SETDEF [DISPLAY] | NO DISPLAY]

SETDEF [PAGE NO PAGE]

Explanation: The SETDEF command lets you display or define the disk search order, the temporary drive, and the filetype search order. The SETDEF definitions affect only the loading of programs and/or execution of SUBMIT (SUB) files. The SETDEF command also lets you turn on/off the DISPLAY and PAGE modes for the system. When DISPLAY mode is on, the system displays the location and name of programs loaded or SUB files executed. When PAGE mode is on, CP/M 3 utilities stop after displaying one full screen of information. Press any key to continue the display.

The system usually searches the specified drive or the default drive for files. The user can use the SETDEF command, to extend the search for program files and submit files, for execution purposes only.

Note: A CP/M 3 program file has a filetype of COM. A file containing commands to be executed by SUBMIT has a filetype of SUB.

Display the Program Loading Search Definitions

Syntax: SETDEF

Explanation: The preceding form of the SETDEF command displays the disk search order, the temporary drive, and the filetype search order.

Drive Search Path:

1st Drive	– default
Search Order	– COM
Temporary Drive	– default
Console Page Mode	– on
Program Name Display	– off

Assign the Drive for Temporary Files

Syntax: SETDEF [TEMPORARY=D:]

Explanation: The preceding form of the SETDEF command defines the disk drive to be used for temporary files. The default drive used for temporary files is the system default drive.

Example: A > SETDEF [TEMPORARY=C:]

The preceding command sets disk drive C as the drive to be used for temporary files.

Define the Disk Drive Search Order

Syntax: SETDEF {d: {,d: {,d: {,d: }}}}

Explanation: The preceding form of the SETDEF command defines the disks to be used by the system for programs and/or submit files to be executed. The CP/M 3 default is to be included in the drive search order.

Example: A > SETDEF C:,*

The preceding example tells the system to search for a program on drive C, then, if not found, search for it on the default drive.

Define the Filetype Search Order

Syntax: SETDEF [ORDER = {typ {,typ})]

where typ = COM or SUB

Explanation: The preceding form of the SETDEF command defines the filetype search order to be used by system for program loading. The filetype, indicated as typ in the syntax line, must be COM or SUB. The CP/M 3 default search is for COM files only.

Example: A > SETDEF [ORDER=(SUB, COM)]

The preceding command instructs the system to search for a SUB file to execute. If no SUB file is found, search for a COM file.

Turn On/Off System Display Mode

Syntax: SETDEF [DISPLAY | NO DISPLAY]

Explanation: The preceding command turns the system display mode on or off. The default system display mode is off. When the display mode is on, CP/M 3 displays the following information about a program file before loading it for execution: drive, filename, filetype (if any), and user number (if not the default user number).

Example: A > SETDEF [DISPLAY]

The preceding command turns on the system display mode. The system now displays the name and location of programs loaded or submit files executed. For example, if you enter the PIP command after turning on the system display mode, CP/M 3 displays the following:

```
A > PIP
A > PIP COM
CP/M 3 PIP VERSION 3.0
```

indicating that the file PIP.COM was loaded from drive A under the current user number. If the current user number is not 0, and if PIP.COM does not exist under the current user number, then the system displays the location of PIP.COM as follows:

```
4A > PIP
A:PIP
CP/M 3 PIP VERSION 3.0
```

indicating that PIP.COM was loaded from drive A under user number 0. This mode is effect until you enter

SETDEF [NO DISPLAY]

to turn off the system DISPLAY mode.

Turn On/Off System Page Mode

Syntax: SETDEF [PAGE NO PAGE]

Explanation: The preceding command turns on/off the system page mode. When the PAGE mode is set to on, CP/M 3 utilities stop after displaying one full screen of information, called a console page. The utilities resume after you press any key.

The default setting of the system page mode is ON.

Example: A > SETDEF [NO PAGE]

The preceding command turns off the system page mode. CP/M 3 utilities do not pause after displaying a full console page, but continue to scroll.

The SHOW Command

Syntax: SHOW {d:[SPACE | LABEL | USERS | DIR |DRIVE]}

Explanation: The SHOW command displays the following disk drive information:

- access mode and amount of free disk space
- disk label
- current user number
- number of files for each user number on the disk
- number of free directory entries for the disk
- drive characteristics

Display Access Mode and Disk Space Available

Syntax: SHOW {d:} {[SPACE]}

Explanation: The preceding form of the SHOW command displays the drive, the access mode for that drive, and the remaining space in kilobytes for the specified drive. SHOW by itself displays the information for all logged–in drives in the system.

Example:

```
A > SHOW B:
B: RW, Space: 18K
A > SHOW
A: RW, Space: 4K
B: RW, Space: 18K
```

The first example shows that drive B has Read–Write access and it has bytes of space left. The second example shows that drive A also is Read–Write and has only 4k bytes left and drive B is Read–Write and has 18k bytes left.

Display Disk Label

Syntax: SHOW {d:} [LABEL]

Explanation: The preceding form of the SHOW command displays disk label information.

Example: A > SHOW B: [LABEL]

The preceding command displays the following for drive B:

Label for drive B:

Directory Label	Passwds Reqd	Stamp Create	Stamp Update	Label	Created	Label	Updat
TOMSDISK	on	on	on	07/04/84	10:30	07/08/84	09.30

The first column, directory label, displays the name assigned to that drive directory. The second column, Passwds Reqd, shows that password protection has been turn on for that drive.

The rest of the columns are currently not used.

Display User Number Information

Syntax: SHOW {d:} [USERS]

Explanation: The preceding command displays the current user number and all the users on the drive and the corresponding number of files assigned to them.

Example: A > SHOW A: [USER]
Active User : 1
Active Files: 0 2 3 4
A: # of files: 95 40 1 26

A: Number of free directory entries: 350
A >

Display Number of Free Directory Entries

Syntax: SHOW {d:} [DIR]

Explanation: The preceding command displays the number of free directory enteries on the specified drive.

Example: A > SHOW C: [DIR]
C: Number of free directory entries: 24

A >

The preceding comamnd show that there are only 24 free directory entries on drive C.

Display Drive Characteristics

Syntax: SHOW {d:} [DRIVE]

Explanation: The preceding form of the SHOW command displays the drive characteristics of the specified drive.

Example: A > SHOW [DRIVE]

The preceding is an example of the system display for the preceding command:

A: Drive Characteristics
1,184: 128 Byte Record Capacity
148: Kilobyte Drive Capacity
64: 32 Byte Directory Enteries
48: Checked Directory Enteries
128: Records / Directory Entry
8: Records / Block
32: Sectors / Track
3: Reserved Tracks
256: Bytes / Physical Record

The SUBMIT Command

Syntax: SUBMIT {filespec} {argument} ... [argument]

Explanation: The SUBMIT command lets you execute a group or batch of commands from a SUB file, which is a file with filetype of SUB.

Usually, you enter commands one line at a time. If you must enter the same sequence of commands several times, you might find it easier to batch the commands together using the SUBMIT command. To do this, create a file and enter your commands in this file. The file is identified by the filename, and must have a filetype of SUB. When you issue the SUBMIT command, SUBMIT reads the file named by the filespec and prepares it for interpretation by CP/M 3. When the preparation is complete, SUBMIT sends the file to CP/M 3 line–by–line, as if you were typing each command.

The SUBMIT command executes the commands from a SUB file as if you are entering the command from the keyboard.

It can contain CP/M 3 commands, nested SUBMIT commands, and input data for a CP/M 3 command or program.

You can pass arguments to SUB files when you execute them. Each argument you enter is assigned to a parameter in the SUB file. The first argument replaces every occurrence of $1 in the file, the second argument replaces parameter $2, etc., up to parameter $9. For example, if your file START.SUB contains the following commands:

```
ERA $1.BAK
DIR $1
PIP $1=A:$2.COM
```

and you enter the following SUBMIT command:

```
A > SUBMIT START SAM TEX
```

the argument SAM is substituted to every $1 in the START.SUB file, and TEX for every occurrence of $2 in the START.SUB file. SUBMIT then creates a file with the parameter substitutions and executes this file. This file now contains the following commands:

```
ERA SAM.BAK
DIR SAM
PIP SAM=A:TEX.COM
```

If you enter fewer arguments in the SUBMIT command than parameters in the SUB file, the remaining parameters are not included in the commands.

If you enter more arguments in the SUBMIT command than parameters in the SUB file, the remaining arguments are ignored.

To include an actual dollar sign, $ in your SUB file, type two dollar signs, $$. SUBMIT replaces them with a single dollar sign when it substitutes an argument for a parameter in the SUB file.

Program Input Lines in a SUB File

A SUB file can contain program input lines. Any program input is preceded by a less than sign, <, as in the following example:

```
PIP
< B:=*.ASM
< CON:= DUMP.ASM
<
DIR
```

The three lines after PIP are input lines to the PIP command. The third line consists only of the < sign, indicating a carriage return. The carriage return causes PIP to return to the system to execute the final DIR command.

If the program terminates before using all of the input, SUBMIT ignores the excess input lines and displays the following warning message:

Warning: Program input ignored

If the program requires more input than is in the SUB file, it expects you to enter the remaining input from the keyboard.

You can enter control characters in a SUB file by using the usual convention of preceding the control character by an up–arrow character,^ , followed by the letter to be converted to a control character. To enter an actual ^ character, use the combination ^ ^. This combination translates to a single ^ in the same manner that $$ translates to a single $.

The SUB File

The SUB file can contain the following types of lines:

- Any valid CP/M 3 command
- Any valid CP/M 3 command with SUBMIT parameters
- Any data input line
- Any program input line with parameters ($0 to $9)

CP/M 3 command lines cannot exceed 128 characters.

Example: The following lines illustrate the variety of lines that can be entered in a SUB file:

```
DIR
DIR *.BAK
PIP LST:=$1.PRN[T$2 $3 $5]
DIR *.ASM
PIP
<B:=*.ASM
<CON:=DUMP.ASM
<
DIR B:
```

Executing the SUBMIT Command

Syntax:

SUBMIT
SUBMIT filespec
SUBMIT filespec argument ... argument

If you enter only SUBMIT, the sytem prompts for the rest of the command. You enter the filespec and arguments.

Example:

A > SUBMIT

The system displays the following prompt. Enter filespec and arguments here, such as:

Enter File to Submit: START B TEX

Another example could be

A > SUBMIT AA ZZ SZ

where AA is the SUB file AA.SUB, ZZ is the argument to replace any occurrences of $1 in the AA.SUB file and SZ is the argument to replace all occurrences of $2 in the AA.SUB file.

The PROFILE.SUB Start–up File

Every time you turn on or reset your computer, CP/M 3 automatically looks for a special SUB file named PROFILE.SUB to execute. If it does not exist, then CP/M 3 resumes normal operation. If the PROFILE.SUB file exists, the system executes the commands in the file. This file is convenient to use if you regularly execute a set of commands before you do your regular session on the computer. For example, if you want a directory of the disk each time you boot up the system, you can create the PROFILE-.SUB file, with the text editor, and enter the DIR command as follows:–

DIR

Then, whenever you bring up the system, the system executes the DIR command. By using this facility, you can be sure to execute a regular sequence of commands before starting your usual session.

The TYPE Command

Syntax: TYPE {filespec {[PAGE]} | [NO PAGE]}

Explanation: The TYPE command displays the contents of an ASCII character file on your screen. The PAGE option displays the console listing in paged mode, which means that the console listing stops automatically after listing n lines of text, where n is usually the system default of 24 lines per page. (See the DEVICE command to set n to a different value). Press any character to continue listing another n lines of text. Press CTRL–C to exit back to the system. PAGE is the default mode.

The NO PAGE option displays the console listing continuously.

If you do no enter a file specification in the TYPE command the system prompts for a filename with the message:

Enter filename:

Respond with the filespec of the file you want to list.

Tab characters occurring in the file named by the file specification are expanded to every eighth column position of your screen.

At any time during the display, you can interrupt the listing by pressing CTRL–S. Press CTRL–Q to resume the listing.

Press CTRL–C to exit back to the system.

Make sure the file specification identifies a file containing character data.

To list the file at the printer and on the screen, type a CTRL–P before entering the TYPE command line. To stop echoing console output at the printer, type a second CTRL–P. The TYPE command displays the contents of the file until the screen is filled. It then pauses until you press any key to continue the display.

Examples: A > TYPE MYPROG.PLI

This command displays the contents of the file MYPROG.PLI on your screen twenty–four lines at a time.

A > TYPE B:THISFILE [NO PAGE]

This command continuously displays the contents of the file THISFILE from drive B on your screen.

SECTION 10 DOS VERSION 3.3

Contents **Page**

INTRODUCTION

In addition to the CP/M Operating System, the LINGO PC–128 mkII also support APPLE DOS. This requires the installation of a user–supplied BASIC Rom–card in the ROM slot. Refer to Section 2 on How to insert Basic Rom Card for installation procedures.

HOW TO BOOT DOS 3.3

IF YOUR COMPUTER IS OFF (COLD–START):

1. Turn on your monitor or TV.

2. Position the system unit power switch to ON.

3. The drive A "IN USE" light will be on to indicate that the computer is ready to access the disk.

4. Insert DOS 3.3 System Master diskette in Drive A and close the drive door.

5. The following will appear on your screen.

DOS VERSION 3.3 SYSTEM MASTER
JANUARY 1,1983

LOADING INTEGER BASIC
INTO MEMORY

6. When the system prompt (]) and cursor appear, this indicate that the booting process has completed.

IF YOUR COMPUTER IS ON (WARM–START):

1. Insert DOS 3.3 System Master diskette in drive A.

2. Type PR#6 after the prompt character (]) and hit RETRN. Alternatively, you may use the programmed function key (F6) to type in this command.

3. The drive A "IN USE" light will be on to indicate that the computer is ready to access the disk.

4. Close the drive door.

5. DOS and Integer Basic will be loaded when the prompt and cursor appears

HOW TO MAKE COPIES OF YOUR DOS 3.3 DISKETTES

The diskette which you want to back–up is known as the "original" diskette. The diskette which you want to copy onto is known as the "duplicate" diskette.

It is a good idea to put a tab on the write–protect notch of your original diskette to ensure that it is not accidentally written over.

To make backup copies, use the copy program COPYA, which is in the DOS 3.3 System disk.

1. Boot up the DOS 3.3 System Master diskette.

2. Insert the duplicate diskette into drive B.

3. Type:
   RUN COPYA
   and press the RETRN key.

4. After a while, you should see this message:

   APPLE DISKETTE DUPLICATE PROGRAM
   ORIGINAL SLOT:DEFAULT=6

5. To use the program's default slot number, slot #6 in this example, just press the RETRN key. (Slot #6 is the slot number of the drive controller card).

6. When the message

   DRIVE:DEFAULT=1

   appears, press RETRN.

7. Press RETRN again when you see

   DUPLICATE SLOT:DEFAULT=6

8. When you see the default drive number for the duplicate diskette:

   DRIVE:DEFAULT=2

   type 1, if you have 1 drive, otherwise press RETRN.

9. The message

   –PRESS 'RETURN' KEY TO BEGIN COPY–

   will appear. Remove the System Master diskette from the drive and insert the original diskette, from which you wish to copy from. (Did you remember to write–protect your original?)

10. When copying is completed you'll see the message

    DO YOU WISH TO MAKE ANOTHER COPY?

    If you type Y (yes), the copying procedure will be repeated, assuming the same drives for the original and duplicate diskettes. Make sure you re–insert the original diskette when you see

    –PRESS "RETURN" KEY TO BEGIN COPY–

    and that you use another diskette (either a blank or one with unwanted information on it) as the duplicate. If you don't wish to make another copy, type N. You will leave the program and DOS will assume your default disk drive to be the one which was the location of the duplicate diskette.

DOS 3.3 COMMAND SUMMARY

This section explains briefly the various commonly used commands under DOS 3.3. For more information please refer to APPLE II DOS MANUAL.

INIT

The INIT command formats a diskette and builds a directory to hold information about the files that will eventually be written to it. You must INIT a blank diskette before using it. INIT will also destroy the old information on a used diskette.

The INIT command requires the use of a BASIC program called the "greeting" program. Each time you boot the diskette, the program will be run automatically. The greeting program is commonly named "HELLO", but you could call it whatever you like.

Here is a step–by–step guide to INITialising a diskette:
(We assume that DOS is already booted)

1. Remove the System Master from your disk drive and replace it with a blank diskette.

2. Type NEW, and then type a greeting program. Here is a simple sample of a greeting program:–

```
10 PRINT "DISKETTE CREATED ON LINGO PC−128 mkII"
20 PRINT "BY PETER ON 1 JAN 1985"
30 END
```

3. Run the program to see if it works by typing "RUN".

4. Once the program is satisfactory, type

   ]INIT HELLO

   When you press the RETRN key, the disk drive "IN USE" light will be on for nearly one minute.

5. When the disk drive "IN USE" light is off, the initializing process is completed. Try booting the diskette, the following message should appear on your screen.

```
DISKETTE CREATED ON LINGO PC–128 mkII
BY PETER ON 1 JAN 1985
```

CATALOG

Use the CATALOG command to list the files on the DOS 3.3 System Master diskette. (You can also use the function key F2). To catalog drive B, type CATALOG,D2 or use the function key F4

Type:

CATALOG

and press the RETRN key.

The following will appear on your screen.

```
]CATALOG

DISK VOLUME 001

*A 003  HELLO
*I 003  APPLESOFT
*B 006  LOADER.OBJ0
*B 042  FPBASIC
*B 042  INTBASIC
*A 003  MASTER
*B 009  MASTER CREATE
*I 009  COPY
*B 003  COPY.OBJ0
*A 009  COPYA
*B 003  CHAIN
*A 014  RENUMBER
*B 020  FID
*A 003  CONVERT13
*B 027  MUFFIN
*B 007  BOOT13
*I 026  APPLEVISION
*I 017  BIORHYTHM
*A 006  BRIAN'S THEME
*I 009  COLOR DEMO
*A 028  LITTLE BRICK OUT
*A 051  PHONE LIST
```

DOS FILES

For each file listed under CATALOG, the type of data ,whether or not the file is locked(protected), the number of sectors the file occupies, and the file name. The filetype is indicated by one of the following:

A – Applesoft BASIC file
I – Integer BASIC file
B – BINARY file
T – Text file

Example of file listing:–

*	A	009	COLOR DEMOSOFT
↑	↑	↑	↑
indicates the file is locked	indicates the file type is an Applesoft BASIC file	indicates the program occupies 9 sectors on the disk	is the program name

LOAD

The LOAD command transfers BASIC program from the diskette into the main memory. To load a program, type the LOAD command, followed by the file name as shown:

LOAD COLOR DEMOSOFT
↑
File Name

RUN

The RUN command loads and runs a BASIC program from a diskette.For example:

RUN COLOR DEMOSOFT
↑
File Name

BRUN

The BRUN command is used to execute a binary or machine language program file. Binary file can be recognized by a symbol 'B' attached to the file when it is listed under CATALOG.

BRUN FID
↑
Binary file

SAVE

The SAVE command stores programs on a diskette. A typical SAVE command will look like this:

]SAVE LINGO
↑
File Name

You will need to type a file name after the SAVE command. If you use a file name that is already on the diskette, whatever program you are saving will replace the program with the same name that was stored on the diskette. Thus the old program is automatically erased. This works as long as the old program and the new program are in the same version of BASIC. If they are not, the new program will not be saved. Instead, the message FILE TYPE MISMATCH appears.

DELETE

The DELETE command enables you to removes file that you no longer need. Removing old files from a diskette makes room for new files. To DELETE a file, type the DELETE command, followed by the filename as shown:

]DELETE APPLE
↑
File Name

RENAME

The RENAME command enables you to change a file's name.

```
]RENAME   APPLE,LINGO
            ↑       ↑
   Old filename   New filename
```

LOCK

Some programs or data files on a diskette must be kept permanently. For this purpose, DOS supports a protective technique called file locking. Locking a file prevents it from being accidentally deleted or written over. To lock a file, enter the LOCK command, followed by the file name.

```
]LOCK LINGO
       ↑
   File Name
```

UNLOCK – Unlocking The Lock File

When you decide to write over or delete a locked file, you can remove the lock with the command UNLOCK, followed by the file name, as shown below:

```
]UNLOCK LINGO
         ↑
   File Name
```

TECHNICAL REFERENCE

SECTION 11 SYSTEM ORGANISATION

Contents **Page**

OVERVIEW

The LINGO PC–128 mkII is designed primarily to encompass all the necessary requirements of a personal computer system. The System Unit is the standalone tabletop unit that contains the power supply, the drives and the system board.

SYSTEM BOARD

The System board fits horizontally in the base of the System unit and is approximately 33cm by 24cm. It is a multilayer computer–aided–design board, with high noise immunity.

DC power from the power supply enter the board through a 6–pin connector. Other connectors on the board are for attaching the keyboard, and the two drives. Five 50–pin card edge–sockets are also mounted on the board.

The System board consists of the functional areas as shown in the block diagram:–

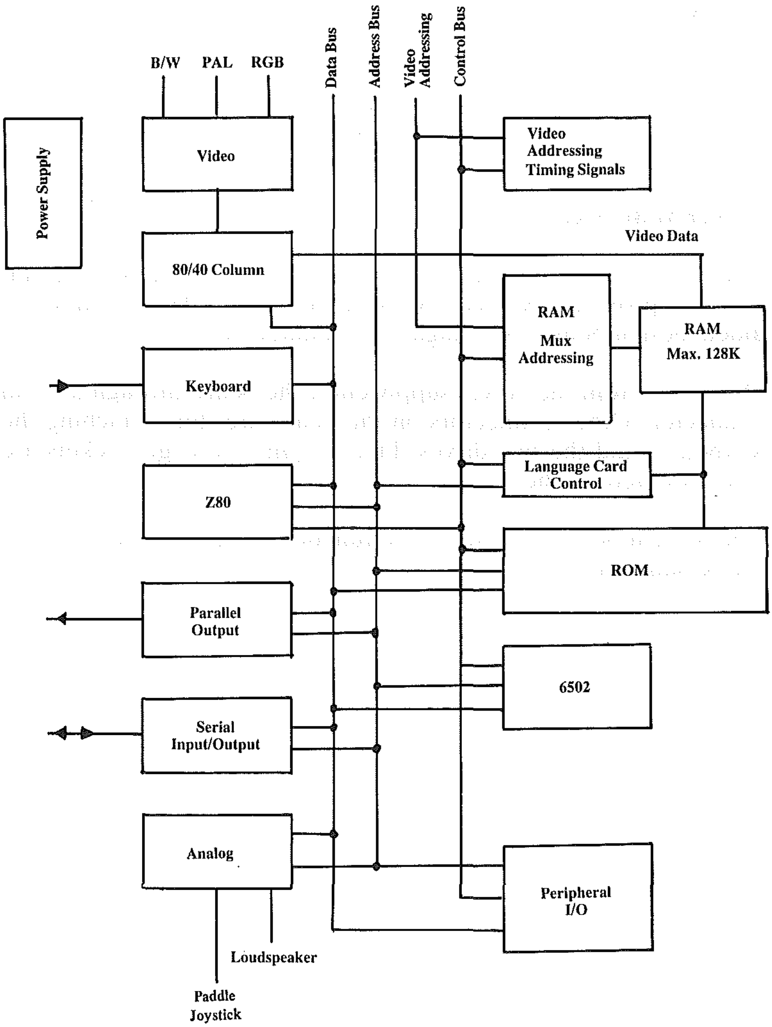


FIG 11–1 LINGO PC–128 mkII Block Diagram

TIMING

System timing is derived from the 14.318 MHz crystal on the system board. This signal is divided and further used to generate all remaining timing signals except for the 4.4336518 MHz Color Reference signal for the PAL video output. The 4.4336518 MHz signal is obtained from another crystal.

Timing Signals

SIGNAL	DESCRIPTION
14M	Master oscillator output. Used to derive other timing signals
7M	7.159 Mhz timing signal
ϕO	1.023 MHz Phase O system clock. Compliment to ϕ1. Sometimes referred to as ϕ 2 in other literature.
ϕ1	1.023 MHz Phase 1 system clock. Compliment to ϕO.
Q3	General purpose timing signal. Twice the frequency of the system clocks, but asymmetrical.

Video control and addressing signal are also generated by this circuitry. Video generation consists of 192 scan lines on the video screen. These are grouped into 24 lines of eight scan lines each. Each scan line can display up to 80 bytes of memory. Fifteen synchronization signals are used. They consist of "H" (horizontal) and "V" (vertical) groups of signals. The HO through H5 signals are used to define displayed position on a line. The VO through V4 signals are used to define the vertical line position on the screen. The VA through VC signals are used to the vertical scan line position within the vertical screen line.

POWER SUPPLY

The power supply is a high efficiency switching type. Built–in circuitry, provides overcharge and undercharge as well as overload protection. The supply is connected to the system board with a keyed connector. The connector is shown as follows:–

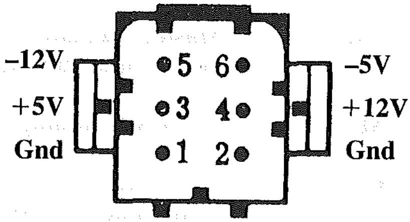


FIG 11–1 Power Supply Connector

SECTION 12 MEMORY ORGANISATION

Contents **Page**

INTRODUCTION

The LINGO PC–128 mkII's 6502 and Z–80 microprocessor can directly reference a total of 65,536 (or 64K) distinct memory locations. You can visualise the system's memory as a book with 256 "pages", with 256 memory locations on each page. For example, "page \$20" is the 256 memory locations beginning at address \$2000 and ending at location \$20FF. Since 6502 and Z–80 uses two. eight–bit bytes to form the address of any memory location, you can think of the higher byte as the page number and the lower byte as the "location within the page".

SYSTEM MEMORY MAP

The system memory map is as shown below. Notice that there are two banks of 64K memory. Since both 6502 and Z–80 are 8–bit microprocessors, therefore, only one bank (Bank0 or Bank1) of memory can be active at any one time.

The RAM address from \$D000 to \$FFFF can be accessed like a language card and is present in both banks of memory, while Text and Graphics screens are available in Bank0 only.

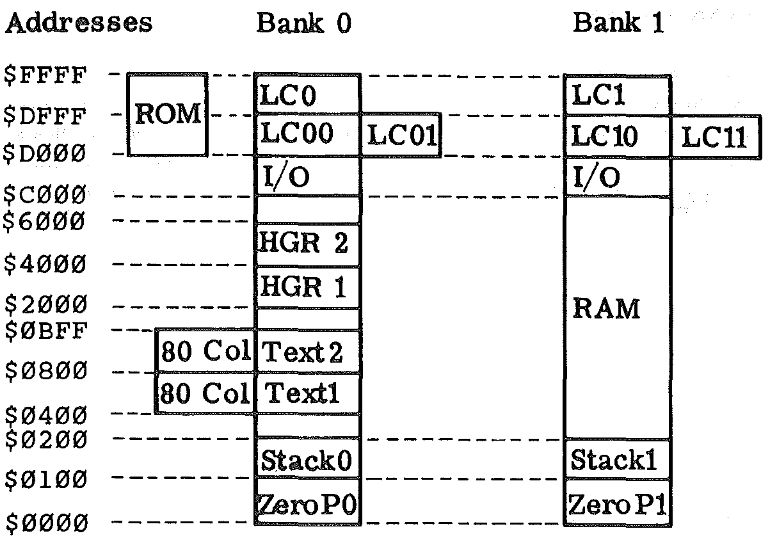


FIG. 12–1 System Memory Map

ON–BOARD MONITOR ROM

The LINGO PC–128 mkII consists of a 2K "system monitor" EPROM residing on the motherboard. This EPROM contain the programs that are executed everytime the LINGO PC–128 mkII is turned on. It also contain the commonly used subroutines which can be "called" by your programs.

The system monitor resides from $F800 to $FFFF in the system memory, and is placed in parallel with the Language Card. To avoid memory conflicts between the EPROM and RAM, the hardware circuitry ensures that either one can be active at any one time.

OPTIONAL ROM–CARD

The address range from $D000 to $FFFF is allocated for the system ROM. The system monitor occupies from location $F800 to $FFFF, but it can be disabled by pulling pin 35 of the ROM slot to 0 volt. Therefore user supplied ROM–cards, containing system monitor and BASIC Interpreters, can be inserted onto the ROM slot.

BANK 0/BANK 1 SWITCHING

The 64K RAM on both banks (bank0 and bank1) is divided into 8K block of memory each. Soft switches are used to make selection of the 8K blocks of memory. One or more of the blocks may be switched at a time, but a block in either Bank 0 or Bank 1, occupying the same memory location, can be active at any one time. The soft switches are located at addresses $C060 to $C06F and are activated by a write operation to these locations. A system reset or cold start will return all 8K blocks of Bank 0 active. The memory bank soft switches are as follows:

Memory Address	Bank 0	Bank 1
$0000 – $1FFF	$C060	$C061
$2000 – $3FFF	$C062	$C063
$4000 – $5FFF	$C064	$C065
$6000 – $7FFF	$C066	$C067
$8000 – $9FFF	$C068	$C069
$A000 – $BFFF	$C06A	$C06B
$D000 – $DFFF	$C06C	$C06D
$E000 – $FFFF	$C06E	$C06F

LANGUAGE CARD AREA

The upper 16K of RAM in both Bank 0 and Bank 1 appears as a Language Card (LC). But, because the memory addresses $C000 through $CFFF are used for the system I/O, only the 12K of memory from $D000 to $FFFF is available to address the 16K of the Language Card. To address all of the 16K of memory on the Language Card, the lower 4K of the Language Card address space is used twice. You can select either of the two 4K memory banks to occupy the space from $D000 to $DFFF at one time. This allows 8K of memory to be accessed in only 4K of address space.

Also, the Language Card shares this memory space with the on–board system ROM or optional Rom–Card. You can ,therefore, choose whether the memory addresses between $D000 and $FFFF will be used to read ROM or read Language Card RAM, and whether these same addresses will be write–enabled or write–protected for Language Card RAM by the use of soft switches.

The Language Card soft switches perform three tasks:–

(1) write–protect or write enable the Language Card RAM
(2) select or de–select RAM read
(3) select page 0 or page 1 for the $D000 – $DFFF address range

These soft switches and its associated functions are as follows:

RAM Selection $D000 – $DFFF		RAM/ROM Selection and RAM write–protect
Page 0	Page 1	
$C080	$C088	Select RAM and write protect it.
$C081	$C089	Select ROM. Two or more successive reads to this address write enables RAM.
$C082	$C08A	Select ROM. Write protect RAM.
$C083	$C08B	Select RAM. Two or more successive reads to this address write enables RAM.

There are two additional soft switches provided for use with this memory. These switches are used to prohibit the switching between Language Card RAM and ROM. Effectively they make the system appear to have no language card area. Writing to \$C00F causes the system not to recognize the \$C08X soft switches. If the write operation is performed when the \$D000 – \$DFFF RAM range is enabled, the switching back to ROM is impossible until a write is performed to \$C00E. Writing to \$C00E or resetting the system causes the \$C08X soft switches to be recognized again.

SECTION 13 VIDEO

Contents **Page**

VIDEO OUTPUTS

The system has three video outputs; Monochrome, modulated PAL color, and RGB outputs which can be used simultaneously. These video outputs are described below:

(1) Monochrome. This output should be used with high resolution monochrome monitors. The video output does not contain the PAL color burst signal, which will appear as noise and cause video degradation on a high quality monochrome monitor. This output is located at the rear panel of the computer, and is labelled as 'B+W/PAL'. For use with PAL monitors, please refer to Chapter 7 for configuring this output to a PAL composite signal.

(2) PAL r.f. This output provides a modulated PAL color signal which can be directly connected to a home TV. This output is available on the rear panel of the computer, labelled as 'HOME TV'.

(3) RGB Output. This output provides analog Red, Green and Blue (RGB) and a composite synchronization signal. When connected to a RGB monitor, this output provides a better color display than the modulated PAL output. The output is available on the rear panel of the computer, labelled as 'RGB'. The pinout of the RGB connector is as shown in FIG 13–1.

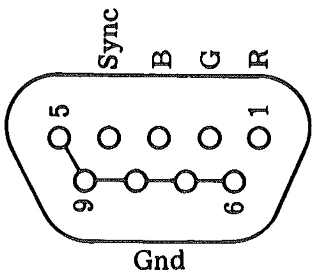


FIG 13–1 RGB Output Pinout

VIDEO MODES

Three different kinds of information can be displayed on the screen:

(1) TEXT. The Lingo PC–128 mkII can display 40 or 80 characters by 24 lines. The characters are formed in a dot matrix 7 dots high and 5 dots wide. There is a one–dot wide space on either side of the character and a one–dot high space above each line.

(2) Low–Resolution Graphics. The LINGO PC–128 mkII can display 1,920 colored squares in an array of 40 blocks wide and 48 blocks high. The color of each block can be selected from a set of sixteen colors. There is no space between blocks, so that any adjacent blocks of the same color look like a single larger block.

(3) High–Resolution Graphics. The LINGO PC–128 mkII can also display colored dots on a matrix 280 dots wide and 192 dots high. The dots are the same size as the dots which make up the Text characters. There are six colors available in the High–Resolution Graphics mode. Each dot on the screen can either be black, white or a color, although not all colors are available for every dot.

When the Lingo PC–128 mkII is displaying a particular information on the screen, it is said to be in that particular mode. Thus, if you see words and numbers on the screen, you can reasonably be assured that your Lingo PC–128 mkII is in Text mode. Similarly, if you see a screen full of multicolored blocks, your computer is probably in Low–Resolution mode. The screen can also have 4 lines of text at the bottom of either type of graphics screen. These four lines replace the lower 8 rows of blocks in Low–Resolution Graphics, leaving a 40 by 40 array. In High–Resolution Graphics, they replace the bottom 32 rows of dots, leaving a 280 by 160 matrix. This 'mixed mode' can be used to display both text and graphics simultaneously, but there is no way to display both graphics modes at the same time.

SCREEN MEMORY

The video display uses information in the system's RAM to generate its display. The value of a single memory location controls the appearance of a certain object on the screen. This object can be a character (Text Mode), two stacked colored blocks (Lo–Res), or a line of seven dots (Hi–Res).

SCREEN PAGES

There are actually two areas of RAM from which each video mode can draw its information. The first area is called the 'primary page' or 'page 1'. The second area is called the 'secondary page' or 'page 2' and is an area of the same size immediately following the first area.

Text and Lo–Resolution Graphics share the same memory area for both the primary and secondary pages. The display RAM addresses for the three modes are as follows :

Screen	Page	Starting Hex	Address Decimal	Ending Hex	Address Decimal
Text/Lo–Res	primary	$400	1024	$7FF	2047
	secondary	$800	2048	$BFF	3071
Hi–Res	primary	$2000	8192	$3FFF	16383
	secondary	$4000	16384	$5FFF	24575

SCREEN SWITCHES

The various display modes, pages and mixes are controlled by 'soft switches'. These soft switches have only two positions (for example on or off, text or graphics) and are called 'soft' because they are controlled by software. A program can activate a switch by referencing the special memory location for that switch. The data which are read from or written to the location are irrelevant; it is the reference to the address of the location which activate the switch.

There are eight special memory locations which control the setting of the soft switches for the various video modes. They are set up in pairs so that when you reference one location of the pair, you turn its corresponding mode 'on' and its companion mode 'off'. The pairs are :

Location: Hex	Decimal	Decimal Equivalent	Description
$C050	49232	–16304	Display a GRAPHICS mode.
$C051	49233	–16303	Display TEXT mode.
$C052	49234	–16302	Display all TEXT or GRAPHICS.
$C053	49235	–16301	Mix TEXT and a GRAPHICS mode.
$C054	49236	–16300	Display primary page (Page 1).
$C055	49237	–16299	Display secondary page (Page 2).
$C056	49238	–16298	Display LO–RES GRAPHICS mode.
$C057	49239	–16297	Display HI–RES GRAPHICS mode.

There are ten distinct combinations of these switches :

Screen	Primary Page: Switches		Secondary Page: Switches	
All text	$C054	$C051	$C055	$C051
All Lo–Res Graphics	$C054 $C052	$C056 $C050	$C055 $C052	$C056 $C050
All Hi–Res Graphics	$C054 $C052	$C057 $C050	$C055 $C052	$C057 $C050
Mixed Text and Lo–Res	$C054 $C053	$C056 $C050	$C055 $C053	$C056 $C050
Mixed Text and Hi–Res	$C054 $C053	$C057 $C050	$C055 $C053	$C057 $C050

To set the Lingo PC–128 mkII into one of these modes, a program needs only to reference to the addresses of the memory locations which corresponds to the switches that set that mode. Machine language programs should use the hexidecimal addresses given above; BASIC programs should PEEK or POKE their decimal equivalents. In switching into one of the Graphics modes, it is advisable to reference the TEXT/GRAPHICS switch last. In this way, all other changes in mode will then take place invisibily behind the text, so that when Graphics mode is set, the finished graphics screen appears all at once.

TEXT MODE

In the Text mode, the Lingo PC–128 mkII can display 24 lines of characters with up to 40 or 80 characters on each line. Each character on the screen represents the contents of one memory location from the memory range of the page being displayed., The primary text page starts at location number 1024 ($0400) and extends to 2047 ($BFF). The secondary text page begins at location number 2048 ($0800) and extends up to location 3071 ($07FF).

Each of these pages is 1024 bytes long, although only 960 characters are displayed on the screen. The remaining 64 bytes in each page which are not displayed on the screen are used as temporary storage locations by peripheral cards. Each I/O slot has been assigned with 8 bytes of temporary storage.

80 COLUMN OPERATION

A very simple method has been implemented for 80 column display.
Static RAM are placed in parallel with the dynamic RAM. The static and dynamic RAM are referenced at the same addresses, but soft switches are used to activate either one. $COOC activates the dynamic RAM while $C00D activates the static RAM.

These two soft switches are used to switch back and forth between the different RAMs. This technique requires that all characters with an even column number be located in dynamic RAM and that all odd column characters be located in static RAM. An additional soft switch is used to activate the hardware to display 80 characters on the screen. The static RAM can still be referenced (read or write) even if the 80 character display is not on. The soft switches and their corresponding functions are listed below:

Location (Hex)	Function
$C00A	80 character display off
$C00B	80 character display on
$C00C	Select dynamic RAM
$C00D	Select static RAM

80 column operation is supported by the software routines in the EPROM at location IC94, and is transparent to the user. The 80 column operation is supported in the following environment :

(1) CP/M. Set the '40/80 COLUMN' switch to the 80 column mode. The screen will automatically switch to 80 column display when any CP/M disk is booted. In order to display in 40 column, switch the computer off. Then toggle the switch labelled '40/80 COLUMN' to the 40 column position. The screen will now display in 40 columns only.

(2) BASIC. Set the '40/80 COLUMN' to the 80 cloumn mode. The 80 column operation can be activated by typing 'PR#3 ‹cr›', and de–activated by pressing the 'RESET' key and 'CTRL' key at the same time.

FIG 13–2 is a memory map of the primary text display area. The map shows the 40 columns and 24 rows as they are displayed in 40 column display. To obtain the address of any one byte, add the row and column of the byte in question. In 80 column display mode, each of the boxes on this map represents two characters, one in dynamic memory and the other in static memory. For secondary text display area add 1024 (hex $400) to the column address.

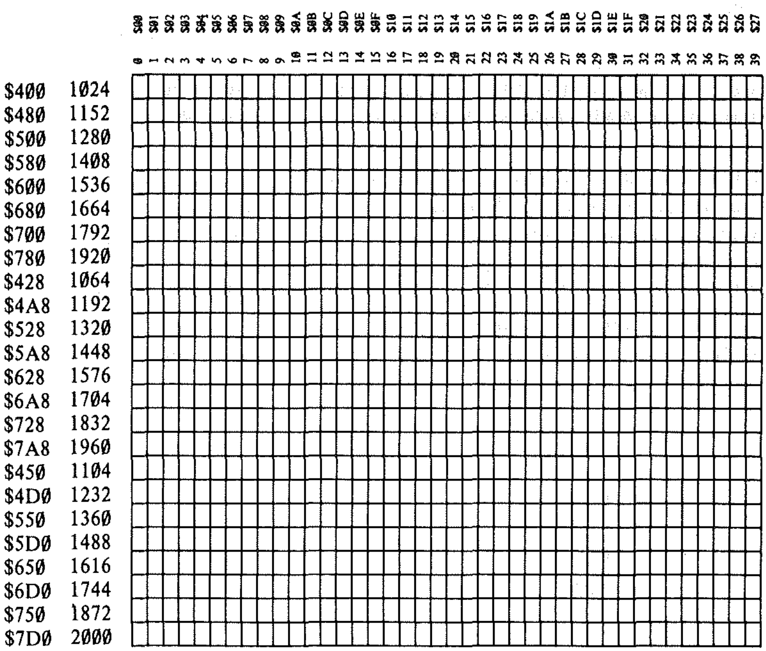


FIG 13–2 Map of the Text Screen

LOW RESOLUTION GRAPHICS

In the Low-Resolution Graphics mode, the Lingo PC-128 mkII uses the same locations in memory used by the Text mode, but in a different format. In this mode, each byte of memory is displayed not as an ASCII character, but as two colored blocks, stacked one on top of the other. The screen can show an array of blocks 40 wide by 48 high (graphics mode), or 40 wide by 40 high with 4 lines of text at the bottom of the screen (mixed graphics and text mode).

Since each byte in the Low-Resolution Graphics represents two blocks on the screen, stacked vertically, each byte is divided into two equal sections, called 'nybbles'. Each nybble can hold a value from 0 to 15. The value which is in the lower nybble of the byte determines the color for the upper block of that byte on the screen, and the value which is in the upper nybble determines the color for the lower block on the screen. The colors available for Low–Resolution Graphics are as follows :

Decimal	Hex	Color
0	$0	Black
1	$1	Magenta
2	$2	Dark Blue
3	$3	Purple
4	$4	Dark Green
5	$5	Grey 1
6	$6	Medium Blue
7	$7	Light Blue
8	$8	Brown
9	$9	Orange
10	$A	Grey 2
11	$B	Pink
12	$C	Light Green
13	$D	Yellow
14	$E	Aquamarine
15	$F	White

So, a byte containing the hexadecimal value $D8 would appear on the screen as a brown block on top of a yellow block. Using decimal arithmetic, the color of the lower block is determined by the quotient of the value of the byte divided by 16; the color of the upper block is determined by the remainder.

FIG 13–3 is a map of the Low–Resolution Graphics screen, with the memory location addresses for each block on the screen.

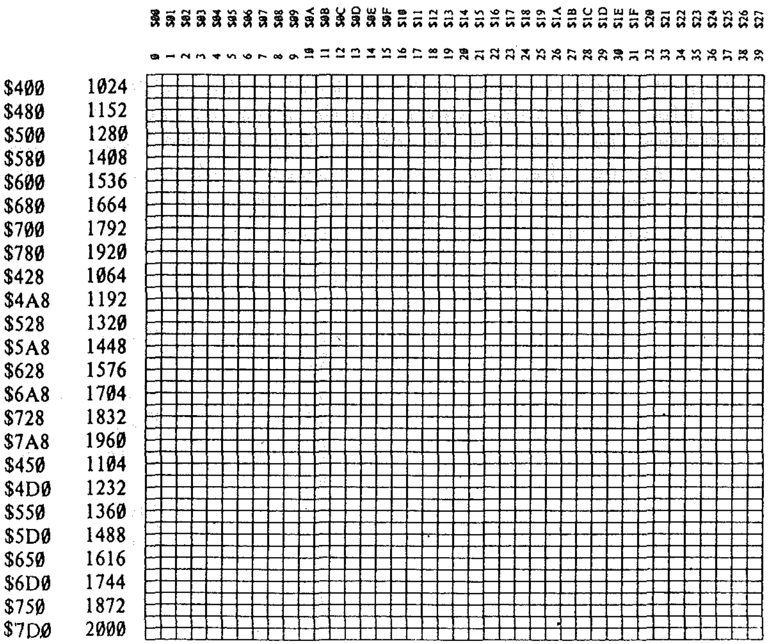


FIG 13–3 Map of the Low–Resolution Graphics Mode

HIGH–RESOLUTION GRAPHICS

When your Lingo PC–128 mkII is in the High–Resolution Graphics mode, it can display 53,760 dots in a matrix 280 dots wide and 192 dots high. When in mixed Hi–Res and Text mode, it can display a matrix 280 dots wide and 160 dots high with 4 lines of text at the bottom of the screen.

The High–Resolution Graphics mode takes its data from an 8,192 byte area of memory, usually called a 'picture buffer'. There are two separate picture buffers; one for the primary page and the other for the secondary page. Both of these buffers are independent and separate from the memory areas used for Text and Low–Resolution Graphics. The primary page picture buffer begins at memory location number 8192 (hex $2000) and extends up to location number 16383 (hex $3FFF). The secondary page picture buffer begins at memory location number 16384 (hex $4000) and extends up to memory location number 24575 (hex $5FFF).

Each dot on the screen represents one bit from the picture buffer. Seven of the eight bits in each byte are displayed on the screen, with the remaining bit used to select the colors of the dots in that byte. The least significant bit (bit 0) of the first byte in the line is displayed on the left edge of the screen, followed by the second bit, then the third, etc. The most significant bit (bit 7) is not displayed. Then follows the first bit of the next byte, and so on. On a black–and–white monitor, the dots whose corresponding bits are "1' appear white; the dots whose corresponding bits are '0' appear black. On a color monitor, it is not so simple. If the bit is '0', its corresponding dot will always be black. If a Bit is '1', however,its color will depend upon its position of that dot on the screen. If the dot is in the leftmost column, called 'column 0, or in any even–numbered column, then it will appear violet. If the dot is in the rightmost column (column 279) or any odd–numbered column, then it will appear green. If two dots are placed side–by–side, they will both appear white. If the undisplayed bit of a byte is turned on, then the colors blue and red are substituted for violet and green, respectively. Thus there are six colors available in the High–Resolution Graphics mode, subjected to the following limitations :

(1) Dots in even columns must be black, violet, or blue.

(2) Dots in odd columns must be black, green, or red.

(3) Each byte must be either a violet/green or blue/red byte. It is not possible to mix green and blue, green and red, violet and blue, or violet and red in the same byte.

(4) Two colored dots side–by–side always appear white, even if they are in different bytes.

FIG 13–4 shows the primary display screen in High–Resolution Graphics mode with the memory addresses of each line on the screen. For the secondary display screen, add 8192 (hex $2000) to the column base address.

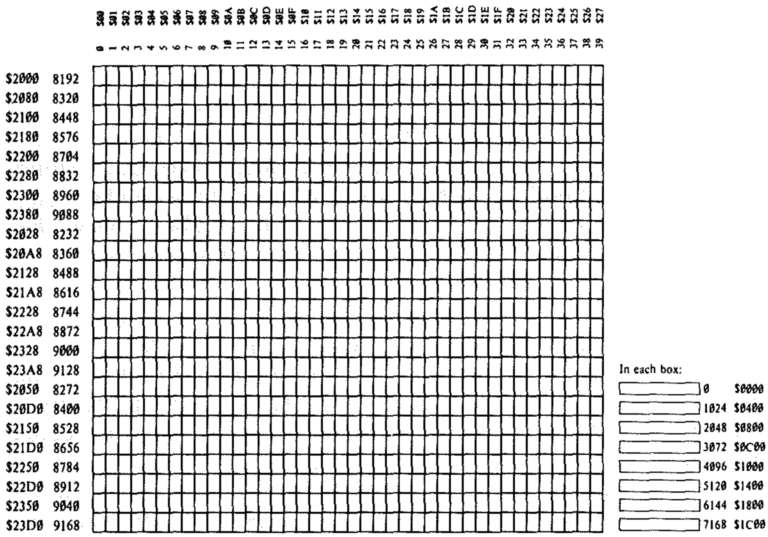


FIG 13–4 Map of the High–Resolution Graphics Screen

SECTION 14 INPUT/OUTPUT ORGANISATION

Contents **Page**

INTRODUCTION

The Lingo PC–128 mkII Input and Output functions can be divided into two categories; those functions which are performed on the motherboard itself, and those functions which are performed by peripheral interface cards plugged into the peripheral expansion 'slots'.

BUILT–IN I/O

The on–board I/O functions are controlled by 128 memory locations in the memory map, beginning at location 49152 ($C000) and extending up to 49279 ($C07F). Ideally, 128 different functions are available, but in some instances, as many as sixteen different functions perform exactly the same function.

THE SPEAKER

A small speaker has been built inside the computer. It is connected such that it can be activated by a program to make various sounds.

The speaker is controlled by a soft switch. The switch can put the paper cone of the speaker in two positions; 'in' and 'out'. This soft switch is not like the soft switches controlling the various video modes, but is instead a 'toggle' switch. Each time a program references the speaker's memory address, the speaker will change state; i.e. change from 'in' to 'out' or vice versa. Each time the state is changed, the speaker produces a tiny 'click'. By referencing the address of the speaker switch frequently and continously, a program can generate a steady tone from the speaker.

The soft switch for the speaker is associated with memory location number 49200 ($C030). Any reference to this address will cause the speaker to emit a click.

A program can reference the address of the special location for the speaker by performing a 'read' or 'write' operation to that address. The data which are read or written are irrelevant, as it is the address which throws the switch. Note that a 'write' operation on the 6502 microprocessor actually performs a 'read' before the 'write', so that if you use a 'write' operation to flip any soft switch, you will actually throw that switch twice. For toggle–type soft switches, such as the speaker switch, this means that a 'write' operation to the special location controlling the switch will leave the switch in the same state it was before the operation was performed.

THE GAME I/O CONNECTOR

The purpose of Game I/O is to allow you to connect special input and output devices, and is normally connected to a joystick or a game paddle, used specifically for game programs. Two connectors are provided at the rear panel; one is a sixteen pin DIP socket and another is a D–9 female connector. Game joystick can be plugged into one of the connectors, but the choice depends on the type of connector used by the joystick. Only one of the Game I/O connector can be used at any one time.

The pinout of the 16–pin DIP socket and the D–9 for the Game I/O connectors are as follows:

16 Pin Socket Pin No.	D–9 Connector Pin No.	Description
1	2	+5v. +5 volt power supply.
2, 3, 4	7, 1	PB0 – PB2 Single–bit (pushbutton) inputs. These are standard 74LS series TTL inputs.
5	––	$C040 STROBE A general–purpose strobe. This line, normally high, goes low during O0 of a read write cycle to any address from $C040 through $C04F. This is a standard 74LS TTL output.
6, 7, 10, 11	5, 8	GC0–GC3 Game controller inputs. These should each be connected through a 150K ohm variable resistor to +5V.
8	3	Gnd System electrial ground.
12 – 15	––	AN0–AN3 Annunciator outputs. These are standard 74LS series TTL outputs and must be buffered if used to drive other than TTL inputs.
9, 16	4, 6, 9	NC No internal connection.

+5v	1	16	NC
PB0	2	15	AN0
PB1	3	14	AN1
PB2	4	13	AN2
$\overline{\text{C040}}$ $\overline{\text{STROBE}}$	5	12	AN3
GC0	6	11	GC3
GC2	7	10	GC1
Gnd	8	9	NC

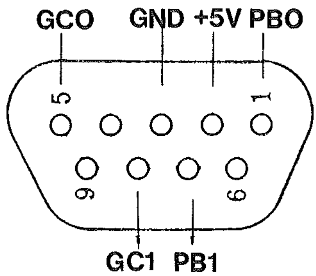


FIG 14–1 Game I/O Connector Pinout

The Game I/O allows you to connect three one–bit inputs, four one–bit outputs, a data strobe, and four analog inputs to the Lingo PC–128 mkII, all of which can be controlled from your programs.

ANNUNCIATOR OUTPUTS

The four one–bit outputs are called 'annunciators'. Each annunciator output can be used as an input to some other electronic device, or the annunciator outputs can be connected to circuits to drive lamps, relays, speakers, etc.

Each annunciator is controlled by a soft switch. The addresses of the soft switches for the annunciator are arranged into four pairs, one pair for each annunciator. If you reference the first address in a pair, you turn the output of its corresponding annunciator 'off'; if you reference the second address in the pair, you turn the annunciator's output 'on'. When an annunciator is 'off', the voltage on its pin on the Game I/O is near 0 volts; when an annunciator is 'on', the voltage is near 5 volts. There are no inherent means to determine the current setting of an annunciator bit. The annunciator soft switches are as follows:

Annunciator	State	Address: Decimal		Hex
0	off	49240	−16296	$C058
	on	49241	−16295	$C059
1	off	49242	−16294	$C05A
	on	49243	−16293	$C05B
2	off	49244	−16292	$C05C
	on	49245	−16291	$C05D
3	off	49246	−16290	$C05E
	on	49247	−16289	$C05F

ONE−BIT INPUTS

The three one−bit inputs can each be connected to either another electronic device or to a push−button. You can read the state of any of the one−bit inputs from a machine language or BASIC program by referencing their special addresses. The locations for the three one−bit inputs have the addresses 49249 ($C061) through 49251 ($C063).

ANALOG INPUTS

The four analog inputs can be connected to 150k ohm variable resistors or potentiometers. The variable resistance between each input and the +5 volt supply is used in a one−shot timing circuit. As the resistance on an input varies, the timing characteristics of its corresponding circuit change accordingly. Machine language programs can sense the changes in the timing loops and obtain a numerical value corresponding to the position of the potentiometer.

Before a program can start to read the setting of a potentiometer, it must first reset the timing circuits. Location number 49264 (−16272 or hex $C070) does just this. When you reset the timing circuits, the values contained in the four locations 49252 (−16284 or hex $C064) through 49255 (−16281 or hex $C067) becomes greater than 128 (bit 7 set). Within 3.060 milliseconds, the values contained in these four locations should drop below 128. The exact time it takes for each location to drop in value is directly proportional to the setting of the game paddle associated with that location.

STROBE OUTPUT

There is an additional output, called the 'strobe', which is normally +5 volts but will drop to zero volts for a duration of one–half microsecond under the control of a machine language or BASIC program. You can trigger this 'strobe' by referencing to location number 49216 (–16320 or hex $C04F). Note that if you perform a 'write' operation to this location, you will trigger the strobe twice.

PERIPHERAL BOARD I/O

Inside the Lingo PC–128 mkII, there is a row of 5 expansion 'slots', numbered as ROM, 4, 5, 6 and 7, starting from the right. These slots allow a variety of peripheral cards to be connected to the LINGO PC–128 mkII. In order to make the peripheral cards simpler and more versatile, the circuitry has allocated a total of 280 byte locations in the memory map for each of the slots. There is also a 2k byte 'common area', which all peripheral cards can share.

PERIPHERAL CARD I/O SPACE

Each slot is given sixteen locations beginning at location $C080 for general input and output purposes. Each peripheral card can use these locations as it pleases, and can determine when it is being selected by sensing pin 41 (called DEVICE SELECT) on its peripheral connector. Whenever the voltage on this pin drop to 0 volts, the address which the microprocessor is calling is somewhere in that peripheral card's 16–byte allocation. The peripheral card can then look at the bottom four address lines to determine which or its sixteen addresses is being called.

PERIPHERAL CARD ROM SPACE

Each peripheral slot has been reserved with one 256–byte page of memory. This page is usually used to contain 256 bytes of ROM or Programmable ROM (PROM), which contains driving programs or subroutines for the peripheral card. In this way, the peripheral interface cards can be 'intelligent'. They contain their own driving software so that you do not need to load separate programs in order to use the interface cards.

The page of memory reserved for each peripheral slot has the page number $Cn, where n is the slot number. The signal Pin 1 (called I/O SELECT) of each peripheral slot will become active (drop from +5 volt to ground) when the microprocessor is referencing an address within that slot's reserved page. Peripheral cards can use this signal to enable their PROMs, and use the lower eight address lines to address each byte in the PROM.

The table below illustrates the addresses allocated to each I/O slot. You will notice that I/O slot connectors 0, 1, 2 and 3 do not exist as they are already being occupied by the motherboard.

Slot No.	Device Select (Pin 41)	I/O Select (Pin 1)	Remarks
0	$C080–$C08F	$C000–$C0FF	Not Available. This slot is used by the on–board 16K Language Card.
1	$C090–$C09F	$C100–$C1FF	Not Available. This slot is used by the on–board printer interface.
2	$C0A0–$C0AF	$C200–$C2FF	Not Available. This slot is used by the on–board serial interface.
3	$C0B0–$C0BF	$C300–$C3FF	Not Available. This slot is used by the on–board 80 column display but may be disabled by removing jumper "JF" on the motherboard, or setting the '40/80 COLUMN' switch to the 40 column mode.
4	$C0C0–$C0CF	$C400–$C4FF	This slot is free for optional expansion card.
5	$C0D0–$C0DF	$C500–$C5FF	This slot is free for optional expansion card.
6	$C0E0–$C0EF	$C600–$C6FF	This slot is normally used by the disk interface card.
7	$C0F0–$C0FF	$C700–$C7FF	This slot is free for optional expansion card.

(Continued)

Slot	Device Select	I/O Select	Remarks
ROM	--	--	This slot is reserved exclusively for optional ROM-card containing system monitor and BASIC interpreter.

EXPANSION ROM

The 2K memory range from location $C800 to $CFFF is reserved for a 2K ROM or PROM on a peripheral card, to hold large programs or driving subroutines. The expansion ROM space also has the advantage of being absolutely located in the system memory map, which gives you more freedom in writing your interface programs.

This PROM space is available to all peripheral slots, and more than one card in your Lingo PC-128 mkII can have an expansion ROM. However, only one expansion ROM can be active at one time.

Each peripheral card's expansion ROM should have a flip-flop to enable it. This flip-flop should be turned "on" by the "device select" signal (the one which enables the 256-byte PROM). This means that the expansion ROM on any card will be partially enabled after you first reference the card it is on. The other enable to the expansion ROM should be the "I/O Strobe" line (Pin 20 on the I/O connector). This line becomes active whenever the microprocessor is referencing a location inside the expansion ROM's domain. When this line becomes active, and the flip-flop has been turned on, then your system is referencing the expansion ROM on this particular board.

A peripheral card 's 256-byte PROM can gain sole access to the expansion ROM space by referring to location $CFFF in its initialization subroutine. This location is a special location, and all peripheral cards should recognize it as a signal to turn their flip-flops "off" and disable their expansion ROMs. Of course, this will also disable the expansion ROM on the card which is trying to grab the ROM space, but the ROM will be enabled again when the microprocessor gets another instruction from the 256-byte driving PROM. Now the expansion ROM is enabled, and its space is clear. The driving subroutines can than jump directly into the programs in the ROM, where they enjoy the 2K of absolutely located memory space. An example of such a Expansion Rom Circuit is as shown.

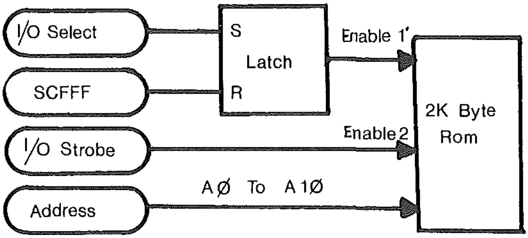


FIG 14–2 Expansion ROM Circuit

PERIPHERAL SLOT SCRATCHPAD RAM

Each of the eight peripheral slots has 8 bytes of RAM allocated to it. These 64 locations are actually in memory pages \$04 to \$07, which are actually reserved for the Text and Low–Resolution Graphics video display. The contents of these locations, however are not displayed on the screen, and are not affected by normal screen operations. The peripheral cards can use these locations for temporary storage of data while the expansion cards are in operation.

These "scratchpad" locations and its corresponding I/O slot have the following addresses. Notice that slots 0, 1,2 and 3 are not available for peripheral cards as they are occupied by the on–board peripherals (16k language card, printer interface, serial interface, and 80 column display).

Slot No.	Scratchpad RAM Address				Remarks
1	\$0479 \$04F9	\$0579 \$05F9	\$0679 \$06F9	\$0779 \$07F9	Not Available. These locations are used by the on–board printer
2	\$047A \$04FA	\$057A \$05FA	\$067A \$06FA	\$077A \$07FA	Not Available. These locations are used by the on–board serial interface.

(Continued)

Slot No.	Scratchpad RAM Address				Remarks
3	$047B $04FB	$057B $05FB	$067B $06FB	$077B $07FB	Not Available. These locations are used by the on–board 80–column display.
4	$047C $04FC	$057C $05FC	$067C $06FC	$077C $07FC	Available. These locations can be used by peripheral cards.
5	$047D $04FD	$057D $05FD	$067D $06FD	$077D $07FD	Available. These locations can be used by peripheral cards.
6	$047E $04FE	$057E $05FE	$067E $06FE	$077E $07FE	Available. These locations can be used be peripheral cards.
7	$047F $04FF	$057F $05FF	$067F $06FF	$077F $07FF	Available. These locations can be used by peripheral cards.

PERIPHERAL CONNECTORS

Inside the Lingo PC–128 mkII computer, there are 5 peripheral I/O connectors. These slots are numbered as ROM, 4, 5, 6, and 7 from the rightmost slot. The ROM slot is specially designed for ROM–Cards and has slightly different signals from the rest of the slots. Peripheral cards designed for the Lingo PC–128 mkII and the Apple II computer can be installed into slots 4, 5, 6, and 7.

Each of the I/O slot consists of a 50–pin connector. The signals at each pin of the connector is as follows:

Pin	Signal	Description
1	I/O	This signal is normally high. It becomes low during Φ0 when a reference is made to $CnXX, where n is the slot number. This signal will drive 10 LSTTL loads.
2–17	A0 – A15	The buffered address bus. The address on these pins becomes valid during Φ1 and remains valid through Φ0. These signals will each drive 5 LSTTL loads.

(Continued)

Pin	Signal	Description
18	R/W	Buffered Read/Write signal. This signal becomes valid the same time the address bus does, and goes high during a read cycle and low during a write cycle. This signal will drive 2 LSTTL loads.
19	N.C.	No Connection.
20	I/O	Normally high, this line goes low during Φ0 Strobe when the address between us contains an address between $C800 and $CFFF. This signal will drive 4 LSTTL loads.
21	RDY	The 6502's RDY input. Pulling this line low during Φ1 will halt the microprocessor, with the address bus holding the address of the current location being fetched.
22	DMA	Pulling this line low disables the 6502's address bus and halts the microprocessor. This line is held high by a 1k ohm resistor to +5V.
23	INT OUT	Daisy−chained interrupt output to lower priority devices. This pin is usually connected to pin 28 (INT IN).
24	DMA OUT	Daisy–chained interrupt output of lower priority devices. This pin is usually connected to pin 22 (DMA IN). DMA OUT for slot 7 is connected to the Z–80
25	+5 volts	+5 volt power supply.
26	INT IN	System electrical ground.
27	DMA IN	Daisy−chained DMA input from higher priority devices. Usually connected to pin 24 (DMA OUT).
28	INT IN	Daisy−chained interrupt input from higher priority devices. Usually connected pin 23 (INT OUT).
29	NMI	Non−Maskable Interrupt. When this line is pulled low, the system begins an interrupt cycle and jumps to the interrupt handling routine at location $3FB.

(Continued)

Pin	Signal	Description
30	IRQ	Interrupt Request. When this line is pulled low, the system begins an interrupt cycle only if the 6502's I (Interrupt disable) flag is not set. If so, the 6502 will jump to the interrupt handling subroutine whose address is stored in locations $3FE and $3FF.
31	RST	When this line is pulled low the microprocessor begins a RESET cycle.
32	INH	When this output line is low, it indicates that all ROMs on the system board are disabled and the Language card is active.
33	−12v	−12 volt power supply.
34	−5v	−5 volt power supply.
35	N.C.	Not connected
36	7M	7MHz clock. This line will drive 2 LSTTL loads.
37	Q3	2MHz asymmetrical clock. This line will drive 2 LSTTL loads.
38	Φ1	Microprocessor's phase one clock. This line will drive 2 LSTTL loads.
39	USER 1	This line, when pulled low, disables all internal I/O address decoding.
40	ΦØ	Microprocessor's phase zero clock. This line will drive 2 LSTTL loads.
41	DEVICE SELECT	This line becomes active (low) on each peripheral connector when the address bus is holding an address between $C0n0 and $C0nF, where n is the slot number plus $8. This line will drive 10 LSTTL loads.
42	D0 − D7	Buffered bidirectional data bus. The data on to this line becomes valid 300nS into Φ0 on a write cycle, and should be stable no less than 100nS before the end of Φ0 on a read cycle. Each data line can drive one LSTTL load.
50	+12v	+12 volt power supply.

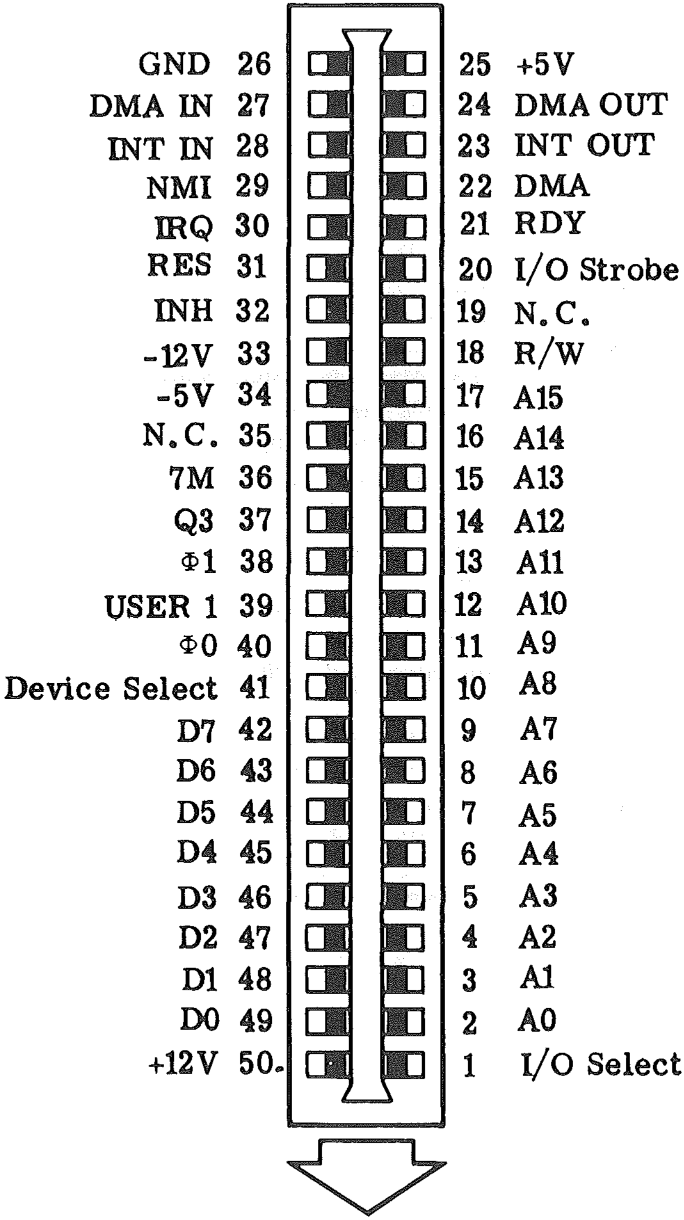


FIG 14–3 Peripheral Connector Pinout

ROM SLOT

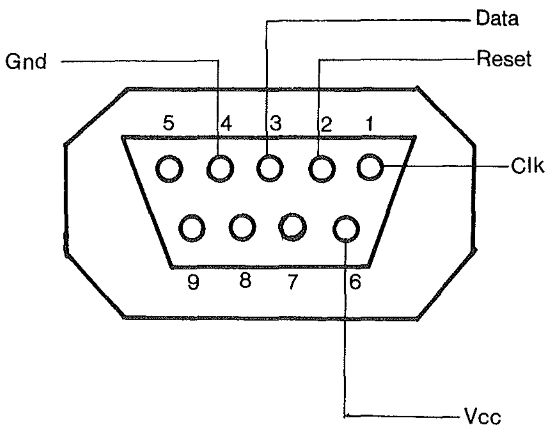
The ROM slot signals are slightly different from the other 4 I/O connectors. The differences are as follows:

Pin	Description
1	No Connection.
27	Decoded address $F000 – $FFFF
28	Decoded address $D000 – $EFFF
41	No Connection.

KEYBOARD SPECIFICATION

The keyboard uses a nine–pin D–type connector and transmits codes serially to the computer, using the following signals:–

PIN	Signal
1	CLK – 9 clk pulses for every ASCII code to be transmitted.
2	RESET - CTRL <RESET> key linked directly to CPU Reset line.
3	DATA – 1 high bit + 8 bits ASCII code.
4	GND
6	+5V



ASCII CODE DEFINITIONS

MINEMONIC	MEANING
SOH	Start of Heading
STX	Start of Text
ETX	End of Text
EOT	End of Transmission
ENQ	Enquiry
ACK	Acknowledge
BEL	Bell
BS	Backspace
HT	Horizontal Tabulation
LT	Linefeed
VT	Vertical Tabulation
FF	Formfeed
CR	Carriage Return
SO	Shift Out
SI	Shift In
DLE	Data Link Escape
DC1	Device Control 1
DC2	Device Control 2
DC3	Device Control 3
DC4	Device Control 4
NAK	Negative Acknowledge
SYN	Sychronous Idle
ETB	End of Transmission Block

MINEMONIC	MEANING
CAN	Cancel
EM	End of Medium
SUB	Subtitute
ESC	Escape
FS	File Separator
GS	Group Separator
RS	Record Separator
US	Unit Separator
DEL	Delete

KEYBOARD KEYSTROKES

HEX	DEC	ASCII	KEY
$00	0	NUL	Control Shift P
$01	1	SOH	Control A
$02	2	STX	Control B
$03	3	ETX	Control C
$04	4	EOT	Control D
$05	5	ENQ	Control E
$06	6	ACK	Control F
$07	7	BEL	Control G
$08	8	BS	←
$09	9	HT	TAB
$0A	10	LF	Control J
$0B	11	VT	Control K
$0C	12	FF	Control L
$0D	13	CR	Control M
$0E	14	SO	Control N
$0F	15	SI	Control O
$10	16	DLE	Control P
$11	17	DC1	Control Q
$12	18	DC2	Control R
$13	19	DC3	Control S
$14	20	DC4	Control T
$15	21	NAK	→
$16	22	SYN	Control V

HEX	DEC	ASCII	KEY
$17	23	ETB	Control W
$18	24	CAN	Control X
$19	25	EM	Control Y
$1A	26	SUB	Control Z
$1B	27	ESC	ESC
$1C	28	FS	Control ;
$1D	29	GS	Control'
$1E	30	RS	Control [
$1F	31	US	Control]
$20	32	SPACE	SPACE
$21	33	!	Shift 1
$22	34	"	Shift'
$23	35	#	Shift 3
$24	36	$	Shift 4
$25	37	%	Shift 5
$26	38	&	Shift 7
$27	39	'	'
$28	40	(	Shift 9
$29	41	)	Shift 0
$2A	42	*	Shift 8
$2B	43	+	Shift =
$2C	44	,	,
$2D	45	–	–
$2E	46	.	.

HEX	DEC	ASCII	KEY
$2F	47	/	/
$30	48	0	0
$31	49	1	1
$32	50	2	2
$33	51	3	3
$34	52	4	4
$35	53	5	5
$36	54	6	6
$37	55	7	7
$38	56	8	8
$39	57	9	9
$3A	58	:	Shift ;
$3B	59	;	;
$3C	60	<	Shift ,
$3D	61	=	=
$3E	62	>	Shift .
$3F	63	?	Shift /
$40	64	@	Shift 2
$41	65	A	Shift A
$42	66	B	Shift B
$43	67	C	Shift C
$44	68	D	Shift D
$45	69	E	Shift E
$46	70	F	Shift F

HEX	DEC	ASCII	KEY
$47	71	G	Shift G
$48	72	H	Shift H
$49	73	I	Shift I
$4A	74	J	Shift J
$4B	75	K	Shift K
$4C	76	L	Shift L
$4D	77	M	Shift M
$4E	78	N	Shift N
$4F	79	O	Shift O
$50	80	P	Shift P
$51	81	Q	Shift Q
$52	82	R	Shift R
$53	83	S	Shift S
$54	84	T	Shift T
$55	85	U	Shift U
$56	86	V	Shift V
$57	87	W	Shift W
$58	88	X	Shift X
$59	89	Y	Shift Y
$5A	90	Z	Shift Z
$5B	91	[	[
$5C	92	\	\
$5D	93	]	]
$5E	94	^	Shift 6

HEX	DEC	ASCII	KEY
$5F	95	-	Shift –
$60	96	'	Control /
$61	97	a	A
$62	98	b	B
$63	99	c	C
$64	100	d	D
$65	101	e	E
$66	102	f	F
$67	103	g	G
$68	104	h	H
$69	105	i	I
$6A	106	j	J
$6B	107	k	K
$6C	108	l	L
$6D	109	m	M
$6E	110	n	N
$6F	111	o	O
$70	112	p	P
$71	113	q	Q
$72	114	r	R
$73	115	s	S
$74	116	t	T
$75	117	u	U
$76	118	v	V

HEX	DEC	ASCII	KEY
$77	119	w	W
$78	120	x	X
$79	121	y	Y
$7A	122	z	Z
$7B	123	{	Shift [
$7C	124	|	Control Shift \
$7D	125	}	Shift]
$7E	126	~	Shift \ ¦
$7F	127	DEL	DELETE

HEX	DEC	STATUS	KEY
$88	136	NUM LOCK OFF	←
$8A	138	NUM LOCK OFF	↓
$8B	139	NUM LOCK OFF	↑
$95	149	NUM LOCK OFF	→
$A1	161	ALT ON	Control F1
$A2	162	ALT ON	Control F2
$A3	163	ALT ON	Control F3
$A4	164	ALT ON	Control F4
$A5	165	ALT ON	Control F5
$A6	166	ALT ON	Control F6
$A7	167	ALT ON	Control F7
$A8	168	ALT ON	Control F8
$A9	169	ALT ON	Control F9
$AA	170	ALT ON	Control F10
$B1	177	ALT ON	Control Shift F1
$B2	178	ALT ON	Control Shift F2
$B3	179	ALT ON	Control Shift F3
$B4	180	ALT ON	Control Shift F4
$B5	181	ALT ON	Control Shift F5
$B6	182	ALT ON	Control Shift F6
$B7	183	ALT ON	Control Shift F7
$B8	184	ALT ON	Control Shift F8
$B9	185	ALT ON	Control Shift F9
$BA	186	ALT ON	Control Shift F10

HEX	DEC	STATUS	KEY
$C1	193	ALT ON	F1
$C2	194	ALT ON	F2
$C3	195	ALT ON	F3
$C4	196	ALT ON	F4
$C5	197	ALT ON	F5
$C6	198	ALT ON	F6
$C7	199	ALT ON	F7
$C8	200	ALT ON	F8
$C9	201	ALT ON	F9
$CA	202	ALT ON	F10
$D1	209	ALT ON	Shift F1
$D2	210	ALT ON	Shift F2
$D3	211	ALT ON	Shift F3
$D4	212	ALT ON	Shift F4
$D5	213	ALT ON	Shift F5
$D6	214	ALT ON	Shift F6
$D7	215	ALT ON	Shift F7
$D8	216	ALT ON	Shift F8
$D9	217	ALT ON	Shift F9
$DA	218	ALT ON	Shift F10

KEYS	CHARACTER STRING	CHARACTER CODES
F1	DIR A: <CR>	$44 49 52 20 41 3A 0D
F2	CATALOG, D1 <CR>	$43 41 54 41 4C 4F 47 2C 44 31 0D
F3	DIR B: <CR>	$44 49 52 20 42 3A 0D
F4	CATALOG, D2 <CR>	$43 41 54 41 4C 4F 47 2C 44 32 0D
F5	STAT *.* <CR>	$53 54 41 54 20 2A 2E 2A 0D
F6	PR#6 <CR>	$50 52 23 36 0D
F7	COPY B:=A: <CR>	$43 4F 50 59 20 42 3A 3D 41 3A 0D
F8	CALL−151 <CR>	$43 41 4C 4C 2D 31 35 31 0D
F9	FORMAT <CR>	$46 4F 52 4D 41 54 0D
F10	INIT HELLO	$46 4E 49 54 20 48 45 4C 4C 4F
PRINT	PRINT	$50 52 49 4E 54

KEYBOARD INTERRUPT MODE

The keyboard can be used in an interrupt mode. When this feature is enabled, the depression of any key causes the generation of an interrupt (IRQ). The keyboard interrupt mode is enabled and disabled by writing to soft switches. The use of this feature requires a user software routine. A list of these locations is shown as follows:–

ADDRESS	FUNCTION
$C008	Keyboard Interrupt OFF
$C009	Keyboard Interrupt ON

SECTION 15 Z–80 MICROPROCESSOR

Contents **Page**

INTRODUCTION

The Z−80 microprocessor is supported by the necessary hardware on the motherboard to permit the direct execution of 8080 and Z−80 programs, including Digital Research's CP/M operating system and all of the programs written to execute in the CP/M software environment.

TIMING

The Z−80 is synchronized and phase locked to the 6502 microprocessor's clock. This is accomplished by generating a clock for the Z−80 from the 6502 clock.

During each video refresh period ($\Phi 1$), the 7 MHz clock is divided into three half clock periods of 135 nanoseconds. The first half−clock is high, the second is low, and the third is high. At the end of the third half clock, the signal goes low and remains low until the start of the next $\Phi 1$. This fourth half clock is normally 563 nsec long. It is stretched by 69 nsec at the end of each video line. This timing scheme creates an effective Z−80 clock rate of 2.041 MHz.

Each type of machine cycle contains one memory access period during $\Phi\emptyset$. The read/write signal is synchronized ensuring that the write can only go low during the time the Z−80 is high. Because all address transitions from the Z−80 occur when the clock is high, they must occur during $\Phi 1$ while the video update accesses are occuring. Thus, each $\Phi\emptyset$ has stable addresses on the bus for the entire duration of the cycle.

Control

The Z−80 is controlled by write commands to the area of memory that normally contains peripheral ROM. It is necessary to use write commands to ensure that the 6502 will not perform two accesses in succession, which would prevent switching back to 6502.

When the power supply is turned on, the RESET signal forces the Z−80 to the off state. The RESET signal is synchronized to the system clock to ensure that a write operation cannot be interrupted. The Z−80 is immediately placed in the WAIT mode and remains in this mode until activated.

Upon receipt of a write to the $C200−$C2FF area of memory, the Z−80 is enabled. The Z−80 remains in the WAIT mode until one memory cycle occurs with Z−80 address information on the bus. At this point the Z−80 is allowed to run with no further wait cycles required. Receipt of another write to the $C200−$C2FF area of memory by the Z−80 will return the Z−80 to the WAIT mode.

ADDRESS BUS INTERFACE

The Z−80 Address Bus interface is constructed such that memory conflicts that exist between the 6502 architecture and the conventions used by both CP/M and the Z−80 microprocessor are resolved. The functions of Dip−switch SW1 on the motherboard are as follows:

SW1 Position				Function
1	2	3	4	
OFF	x	x	x	Add $1000 to all addresses
ON	x	x	x	No translation of addresses
x	OFF	x	x	Disable DMA request to Z−80
x	ON	x	x	Enable DMA request to Z−80
x	x	OFF	x	Disable NMI to Z−80
x	x	ON	x	Enable NMI to Z−80
x	x	x	OFF	Disable IRQ to Z−80
x	x	x	ON	Enable IRQ to Z−80

Switch SW1−1 when off, adds $1000 to the Z−80 addressing. This effectively shifts the Z−80 interrupt addresses and CP/M starting addresses out of the 6502 zero page memory. In addition, addresses in the $C000−$EFFF memory range are shifted to allow apparent contiguous memory for CP/M operation. The following table shows how the memory translator functions when SW1−1 is off:

Z-80 Addressing	6502 Addressing
$0000 – $0FFF	$1000 – $1FFF
$1000 – $1FFF	$2000 – $2FFF
$2000 – $2FFF	$3000 – $3FFF
$3000 – $3FFF	$4000 – $4FFF
$4000 – $4FFF	$5000 – $5FFF
$5000 – $5FFF	$6000 – $6FFF
$6000 – $6FFF	$7000 – $7FFF
$7000 – $7FFF	$8000 – $8FFF
$8000 – $8FFF	$9FFF – $9FFF
$9000 – $9FFF	$A000 – $AFFF
$A000 – $AFFF	$B000 – $BFFF
$B000 – $BFFF	$D000 – $DFFF
$C000 – $CFFF	$E000 – $EFFF
$D000 – $DFFF	$F000 – $FFFF
$E000 – $EFFF	$C000 – $CFFF
$F000 – $FFFF	$0000 – $0FFF

Note that the Z–80 can address contiguous memory from $0000–$DFFF, without addressing the 6502's zero page of memory or the peripheral I/O memory area. When SW1–1 is turned on, the Z–80 addresses memory the same as the 6502 microprocessor.

DMA DAISY CHAIN

The Z–80 microprocessor supports full DMA daisy chain operation. When SW1–2 is on, DMA requests are enabled to interrupt the Z–80. With DMA daisy chain input (pin 27 of I/O slot) driven low, the Z–80 will finish the current machine cycle and then give up control of the bus by raising the DMA control line on pin 22 of the I/O slot. At this time another device may assume control by lowering pin 22. Control must not commence sooner, because the Z–80 buffers will still be driving the bus.

INTERRUPTS

Hardware has been included to allow interrupts to be recognized by the Z–80 as well as the 6502 microprocessors. When SW1–4 is on, the Z–80 will respond to interrupts. The interrupt handler program should not attempt to service the interrupt. Instead, control should be passed back to the 6502 for the actual processing. This permits the 6502, which also sees the interrupt, to clear itself of the interrupt status.

Regardless of the interrupt mode selected for the Z–80, the data byte read during the interrupt sequence will always be $FF. This may be used to vector to a particular memory location for the interrupt handling routine.

Switch SW1–3 performs the same function for the non–maskable interrupt.

SECTION 16 PRINTER INTERFACE

Contents **Page**

INTRODUCTION

The on–board printer interface is a Centronics compatible parallel interface. In addition to advanced text features, it offers high resolution dot graphics (Hi–Res) dump routines residing in firmware. These features are easily invoked by using simple control commands.

DIP SWITCH SETTING

There is no standardization among printer manufacturers for dot graphics. Therefore a DIP switch is provided to define the type of printer that you are using. On the DIP switch (SW3), the positions labelled 1,2,3 and 4 are used by the printer interface.
Set the DIP switches according to the printer that you are using as shown in the table below :

	Dip Switch Positions			
Parallel Printers	1	2	3	4
Epson Printers	*	on	on	on
NEC 8023, C.Itoh 8510	*	on	on	off
Centronics 739–1	*	on	off	on
Anadex Printers, Data South 180	*	on	off	off
Okidata 82 or 83	*	off	on	on

* Note: Position 1 not used.

The parallel interface has many text features, including adjustable margins, line length, page length, etc. All commands and their features are listed in the following pages with a description of their functions. Each one is preceded by a control character which is CTRL–I from BASIC or CTRL–Y from Pascal or CP/M, unless specifically changed by the user's program.

ENTERING CONTROL CHARACTERS

The keyboard 'CTRL' key does not generate a code by itself, but it only alters the codes produced by other keys. For example, to type CTRL−I, press the 'CTRL' key and hold it, and then press the 'I' key. Before you can send a control character to the printer, you must turn on the printer interface by entering PR#1, and then hit RETURN.

For example, CTRL−I S would be entered as folows :

(1) Turn on the parallel interface by entering PR#1, then hit RETURN.

(2) Hold down the 'CTRL' key.

(3) Depress the 'I' key.

(4) Release the 'I' key, then release the 'CTRL' key.

5) Type the 'S' key, then hit 'RETURN' key.

After executing these five steps, the printer will duplicate the present 40 column text screen onto the printer. You will see 'SYNTAX ERROR' printed on the paper and the screen. This is because the BASIC interpreter does not recognise a CTRL command input directly from the keyboard as valid syntax. If you find this 'SYNTAX ERROR' bothersome, enter the command within a program statement. For example, enter

```
10 PR#1: PRINT CHR$(9);"S"
```

and then hit 'RETURN'. Now type 'RUN' and hit 'RETURN'. This will eliminate the 'SYNTAX ERROR'.

TEXT COMMANDS AND FEATURES

Spaces have only been used to make the commands easier to read; do not type the spaces when issuing the commands. The various text commands are listed below :

Command	Function
PR#1	Turns on the parallel interface. All subsequent characters appearing on the video screen wil also be printed on the printer. This command must be used before any of the following commands will be accepted by the printer interface.
PR#0	Turns off the parallel interface. All subsequent data will be sent to the video screen only.
CTRL−I A	Append line feeds onto carriage returns. BASIC will not send a linefeed after a carriage return. This means the print−head will return to the left margin but will not advance the paper, but the parallel interface will automatically issue a linefeed command and advance the paper after every carriage return. If your printer is performing double line spacing, check your printer DIP switches for auto line feed selection. PRINT CHR$(9);"A"
CTRL−I K	Don't append linefeeds onto carriage returns. This feature overrides the CTRL−A command. PRINT CHR$(9);"K"
CTRL−I B	Turns on bell. This allows a CTRL−G to ring the printer bell. Many printers do not have a bell so check your printer manual to see if this feature is available. The bell feature is disabled as a default. PRINT CHR$(9);"B"
CTRL−I C	Turns off bell. PRINT CHR$(9);"C"

Command	Function
CTRL–I H	Allows the high order (bit 8) bit to be output to the printer. On many printers, this has no effect. On some it allows printing of block graphics, line drawing graphics, or special characters (eg. EPSON MX80). However, when block graphics are being printed, the standard character set is not accessible. DIP switch position 1 must be off to allow use of this function. PRINT CHR$(9);"H"
CTRL–I X	Don't output the high order bit to the printer. This puts the printer back into standard text mode and overrides the CTRL–I H command. PRINT CHR$(9);"X"
CTRL–I nN	Set line length to n characters from left side of page (not from left margin). After issuing this command, characters will only be printed on the printer, but not on the screen. A carriage return will automatically be generated after n characters have been printed. If the line length is set to zero, the printer interface will not issue a carriage return until it receives one from the computer. It is recommended that the line length be set to zero when creating dot graphics images independent of the printer interface graphics functions. This is because the Lingo can output as many as a thousands characters per line in the graphics mode. Note : When listing BASIC programs, the printing will automatically be formatted for 40 columns. To print in 80 columns, type the following commands : PRINT CHR$(9);"80N"

CTRL–I I	Transmits characters to both the screen and the printer. When the line length is set with a CTRL–I nN command, the screen will be turned off and printed characters will only go to the printer. If you wish to send characters to the screen as well as the printer, use this command in conjunction with CTRL–I nN. However, this command will not work when listing a program. PRINT CHR$(9);"10L"
CTRL–I nL	Set left margin to nth column. All subsequent printing will have n spaces at the start of each line. For example, when using CTRL–I 10L, all printing will start at the 10th column position. PRINT CHR$(9);"10L"
CTRL–I nR	Set right margin to the nth column from left edge of page (not left margin). This command will prevent words from being spilt at the right margin. The parallel interface will end the line at the first space that occurs after the right margin, but before the end of the line as set by the CTRL–I nN command. A right margin of zero disables this feature. PRINT CHR$(9);"75R"
CTRL–I S	Duplicates the present 40 column text screen on the printer. The screen will be printed 20 spaces from the left edge of the paper. Only forty character screen is printed. Eighty character screen is not supported. PRINT CHR$(9);"S"
CTRL–I 2S	Duplicates the text screen memory located in page 1 and page 2 and prints them side by side for an 80 column output. Once again, eighty character screen is not supported. PRINT CHR$(9);"2S"

CTRL−I nP	Set page length to n. The printer will print six linefeeds after n number of lines have been printed. A page length of zero disables the page length feature. Continuous form paper typically has a total of 66 lines per page (this may vary with the type of printer). Therefore, if you specify a printed page length of 60 lines (CTRL−I 60P), your printer printer will print 60 lines and automatically skip over the perforation with 6 line feeds. PRINT CHR$(9);"60P"
CTRL−I CTRL−Y	Change command character to CTRL−Y. Any control character may be used here instead of CTRL−Y, but you should avoid characters used by your printer or normal text control characters such as CTRL−M (carriage return). PRINT CHR$(9);CHR$(25)
CTRL−Y CTRL−I	Changes back the command character to CTRL−I. PRINT CHR$(25);CHR$(9)

DEFAULT SETTING

The values and conditions listed below are the values which are automatically set any time the parallel interface is activated with a PR#1 command or initialized from PASCAL or CP/M :

Left Margin	0
Right Margin	0
Line Length	0
Page Length	0
Echo to screen	On with BASIC off with PASCAL and CP/M
Carriage Return	Linefeed after carriage return for BASIC. No linefeed after carriage return for PASCAL and CP/M.
Eighth Bit	Off When DIP switch position 1 is on, the bit 8 is not transmitted. When it is on, the bit 8 is under program control (see CTRL−I H).

GRAPHICS FEATURES AND COMMANDS

The parallel interface can output the Hi–Res screen to the printer by using simple commands. All graphics commands consist of the command character (usually CTRL–I) followed by a 'G', and any options described below. If no other options are used (i.e. CTRL–I G carriage return), Hi–Res page 1 will be printed horizontally with every white dot on the screen printed as a black dot on the paper. The different options are listed below :

Command	Function
CTRL–I G 2	Print Hi–Res page 2 instead of page 1. PRINT CHR$(9);"G2"
CTRL–I G S	Print Hi–Res page 1 and 2 side by side. This feature cannot be used in conjunction with rotated or double sized options. Some printers do not have enough character positions to allow this feature to be used (Epson MX–80, Microline 82A). In these cases print two successive rotated pictures (Chart Recorder Mode). PRINT CHR$(9);"GS"
CTRL–I G M	Print the graphics picture and the 4 line text window 20 spaces from the left edge of the paper. The text window may not perfectly align with the picture. This is due to the differences in the graphics and the printer text mode. Rotate, double size, and left margin features do not work with this command. PRINT CHR$(9);"GM"
CTRL–I G D	Print the graphics screen double size. Some printers do not have enough character positions to print a double size image horizontally (eg. MX–80, Microline 82A). If that is the case, you must also use the 'R' option. Otherwise your printer may get confused and 'crash', meaning it will not do anythihg. If that happens you will have to reset the Lingo by pressing the 'reset' key, and then reset the printer by turning it off and then on again. PRINT CHR$(9);"GD"

CTRL–I G E	Prints an 'emphasized' image (Epson MX–80, MX– 80FT, MX–100, NEC 8023 and C.Itoh 8510). When in this mode, the printer will print two closely spaced dots for every one dot that it would normally print. This results in a denser image but printing time is twice as long. PRINT CHR$(9);"GE"
CTRL–I G I	Invert the image before printing. Normally, every white dot on the screen is printed as a black dot on the paper. This works fine for lines and graphs but if you are printing an actual picture of a person or object it will appear like a negative photograph. Using this command, it will print the black postions of the screen as black on the paper, allowing the picture to appear normally. PRINT CHR$(9);"GI"
CTRL–I G L	Print the image at the left margin previously set using the left margin text command. Depending on the type of printer being used, this margin may be affected by the current print density (characters per inch) or may be automatically set to 7 dot positions per character. If this option is not used, the image will be printed in the centre of an eight and a half inch page. PRINT CHR$(9);"10L" PRINT CHR$(9);"GL"
CTRL–I G R	Rotate the picture 90 degrees in a clockwise direction. Some printers require this option when printing the image double size. PRINT CHR$(9);"GR"

When typing control–character commands directly from the keyboard and not from a program, the LINGO PC–128 mkII will give a 'SYNTAX ERROR'. This is normal since the BASIC do not recognise the command. To avoid the 'SYNTAX ERROR', type a one line program. For example : 10 PR#1: PRINT CHR$(9);"GDIR2": PR#0

Then hit RETURN. Now type 'RUN' and hit 'RETURN'.

LOADING HI–RES PAGES 1 AND 2

In order to use the Dual Hi–Res dump command, you must load a binary file into the Hi–Res pages 1 and 2 into the memory. To load a file directly from the keyboard, use the following commands:

For Hi–Res page 1, type : BLOAD filename,A$2000
For Hi–Res page 2, type : BLOAD filename,A$4000

If you wish to load a file from a program, use the following commands:

For Hi–Res Page 1: 10 PRINT CHR$(4);"BLOAD filename,A$2000".
For Hi–Res Page 2: 10 PRINT CHR$(4);"BLOAD filename,a$4000".

The following sample program will load two Hi–Res pictures, one into page 1 and another into page 2, and dump them using the Dual Hi–Res feature.

```
10 PRINT CHR$(4);"BLOAD filename1,A$2000"
20 PRINT CHR$(4);"BLOAD filename2,A$4000"
30 PR#1
40 PRINT CHR$(9);"GS"
60 PRINT PR#0
70 END
```

CHART RECORDER MODE

A chart recorder prints a continous graph on a roll of paper. If you need to chart more information than can be done with a single Hi–Res or Dual Hi–Res screen dump you can simulate a chart recorder by printing successive screens without any intervening spaces. This feature works in both normal and rotated modes on all printers except Centronics 739, which will insert a half linefeed when in the rotated mode. The following programs loads a graph into Hi–Res pages 1 and 2 and dumps them continuously without a space separating the two images :

```
10 PRINT CHR$(4);"BLOAD GRAPH1,A$2000"
20 PRINT CHR$(4);"BLOAD GRAPH2,A$4000"
30 PR#1
40 PRINT CHR$(9);"GR"
50 PRINT CHR$(9);"G2R"
60 PR#0
70 END
```

FOR ADVANCED PROGRAMMERS

The following features are available for advanced (assembly language) programmers :

Command	Function
CTRL−I U	Jump to the user routine starting at $300. This feature is for advanced programmers who wish to bypass the parallel interface firmware. The accumulator will contain the output character. The user routine should 'JMP' to $CF03 when finished, which will restore the stack and registers.
CTRL−I 1U	Jump to the user routine starting at $D000, bank A of RAM card. The parallel interface firmware will enable the RAM card prior to the jump. As with CTRL−I U, The accumulator will contain the output character. The user routine should 'JMP' to $CF00 to disable the RAM card and restore the stack and registers when finished.

To restore the normal parallel interface firmware, you must execute a PR#0 followed by a PR#1.

SPECIAL MEMORY LOCATION

The parallel interface uses those RAM locations set aside for its particular slot and the slot scratchpad area common to all slots. The ROM is turned on using the standard ROM expansion protocol and resides in the locations $C800 to $CFFF. The entry points and the various RAM variables are listed below :

Cold Entry	$C100
Warm Entry	$C102
Left Margin	$479
Right Margin	$679
Line Length	$579
Page Length	$5F9
Character Counter	$4F9
Line Counter	$6F9
Current Cmd. character	$779
Text Flags	$7F9

Text Flag Bit Representations :

Output high order bit	Bit 7
Video turned on	Bit 6
LF after CR	Bit 5
Bell On	Bit 4
Reserved	Bit 0,3

Printer Status Bits:

DB7	Interrupt
DB6	Printer Type
DB5	Printer Type
DB4	Printer Type
DB3	Busy
DB2	Paper Empty
DB1	Select
DB0	Acknowledge

The following locations are used to access the parallel interface:

Address	Read	Write
$C090	Status	Output Port
$C091	Status	Select bank 2 of ROM
$C092	Status	Reset Interrupt Request and IRQ data bit.
$C094	Status	Output/Interrupt on ACK.

Note: A READ in the range of $C100 to $C1FF will select bank 1 of ROM. When an Interrupt (IRQ) has been generated by the parallel interface, bit 7 (MSB) of the status byte will set to '1'.

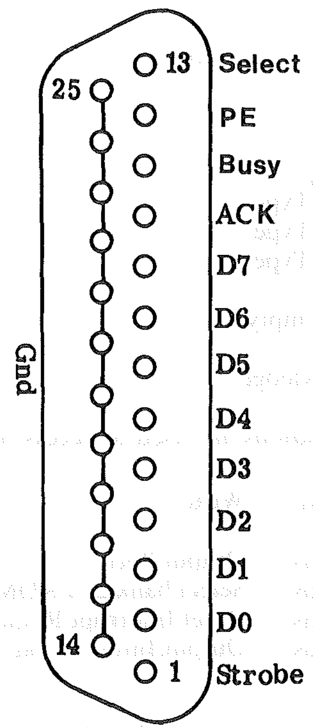


FIG 16–1 Printer Interface Pinout

SECTION 17 SERIAL INTERFACE

Contents **Page**

INTRODUCTION

The on–board serial interface is compatible to the CCS–7710D, and allows the Lingo PC–128 mkII to be connected to peripheral devices such as serial printers, video terminals, and even other computers. When connected to a modem, it allows your computer to communicate through the telephone line.

SERIAL INTERFACE BASIC THEORY

Computer is a very expensive do–nothing unless you can give it data, instruct it what to do with the data, and then have it present the results. To do this, peripheral devices were designed. But because computers could communicate with peripherals in a number of formats, another kind of device was necessary; interfaces. Interfaces translate between a computer, which inputs and outputs in one format, and a peripheral, which inputs and /or outputs in another format.

There are two major types of computer data transfer: parallel and serial. Parallel interfaces transfer data lines simultaneously on parallel data lines. In parallel communications, all data bits must not only be transmitted but also received simultaneously. Simultaneous reception can be assured only if the distance of transmission remains below a certain limit, typically about 3 metres for a computer–to–peripheral cable.

The simultaneous reception problem is not present in serial data transfer, in which the transmitter disassembles words and sends them a bit at a time over a single wire and the receiver reassembles the arriving stream of bits into words.
For this scheme to work, the receiver and transmitter must make identical assumptions about which part of the data stream represents which bit of which word. Two ways have been devised to do this: synchronous and asynchronous modes of transmission. During synchronous transmission, a pre–defined pattern of synchronization bits is sent out first. When the receiver finds this pattern, it divides the subsequent data into pre–defined word–length groups. This requires precise synchronization of transmitter and receiver timing. Asynchronous interfaces handle the problem differently. They add a start bit to the front of each word, plus one or more stop bits to the end. The total group of bits is then transmitted one bit at a time. The receiver can identify the data bits even if it is slightly out of sync with the transmitter because it resynchronizes at the beginning of each word.

In both synchronous and asynchronous modes, the receiver must know how fast the data is being sent. Generally, as long as the receiver expects data at the rate the sender is sending it, the actual rate of transfer does not matter. Common usage has defined many standard signal rates. Usually, they are even multiple of 75 baud «bits per second, including overhead bits such as start and stop bits». A few other rates, 50, 110, and 134.5, are used by the industry giants. The RS–232C standard for serial communication does not specify any standard signal rates, but suggests a practical upper limit of 20 kilobaud and indirecly establishes a theoretical upper limit of 50 kilobaud.

It is not enough for computers and peripherals to be able to exchange data; each must also be able to tell when the other is ready to transmit or receive. This is done with "handshaking" signals. Because a wide variety of handshaking schemes are possible, the EIA created the RS–232C interface specifications to let manufacturers know what to expect. Two "sides" were defined. Because one side of the interface is usually connected to some type of computer terminal, equipment at that end is called Data Terminal Equipment (DTE). Equipment at the other end is called Data Communications Equipment (DCE), because to transmit serial data over long distances via telephone wires a Modulator/ Demodulator, or Modem, is needed. For short distances, modems and telephone wires are not needed, but one side of the interface must be made to think it is DCE.

RS–232C CONFIGURATIONS

RS–232C defines the necessary protocol between the DCE and the DTE for many possible configurations, but of these, only 5 are most commonly used. These are as listed below:

A Transmit Only
B Transmit Only*
C Receive Only
D Half Duplex, or Duplex*
E Full Duplex

Note: The inclusion of Request to Send in a Transmit Function where it would not ordinarily be expected, but could indicate a non–transmit mode to the data communications equipment (DCE) to permit it to remove a line signal or to send synchronizing or framing signals as required.

Configuration A defines a one–way, transmit–only interface, as might be used by a simple serial keyboard. Configuration B is also a one–way, transmit–only interface, but it has more handshaking. A paper tape reader might use this type of interface. Configuration C is also one–way, but receives only, and might be used by a serial printer. With Configuration D, we start getting into two–way traffic. Configuration D is HALF DUPLEX; although it can carry traffic both ways, it can only carry one way at a time. When a modem is used in this configuration, it is often called "two–wire modem" because the telephone is connected with only two wires. This type of interface is not often used. Configuration E is a FULL DUPLEX link. This means that the link can support traffic in both directions at the same time. CRT terminals most often use this type of link. A modem used in this configuration is called a "four–wired modem". The on–board serial interface supports all five of these configurations. In addition, a Data Terminal Ready «DTR» handshake line has been provided. Normally used only on synchronous interfaces, this line is also used by some asynchronous printers.

SETTING THE BAUD RATE

For correct operation of serial transmission, the peripheral devices' baud rate should be set identical to that of the computer. Refer to the peripheral manual on setting of the baud rate. The serial interface baud rate of the Lingo PC–128 mkII can be set by setting the dipswitch SW2, located on the motherboard. The various baud rates and its corresponding switch position are as listed below:

	SW2			
Baud Rate	1	2	3	4
50	ON	OFF	ON	ON
75	OFF	OFF	ON	ON
110	OFF	OFF	OFF	OFF
134.5	ON	ON	OFF	ON
150	ON	OFF	OFF	OFF
200	OFF	ON	OFF	ON
300	OFF	ON	OFF	OFF
600	ON	OFF	OFF	ON
1200	OFF	OFF	ON	OFF
2400	OFF	OFF	OFF	ON
4800	OFF	ON	ON	OFF
9600	ON	ON	ON	OFF
19200	ON	ON	ON	ON
External Clk	OFF	ON	ON	ON

THE PERIPHERAL CABLE

A cable is required to connect your peripheral devices to the computer. In most cases a standard RS−232C connector will be required, but you may have to make some modifications depending on your peripheral's handshaking.

The RS−232C standard for serial data communication defines the characteristics of the interchange signals between Data Terminal Equipment (e.g. Lingo PC−128) and Data Communications Equipment «e.g. modem, serial printer». Each equipment has its own set of handshaking signals that are assigned to specified pins of a DB−25 connector. The possible cable connections for DTE and DCE are as follows:

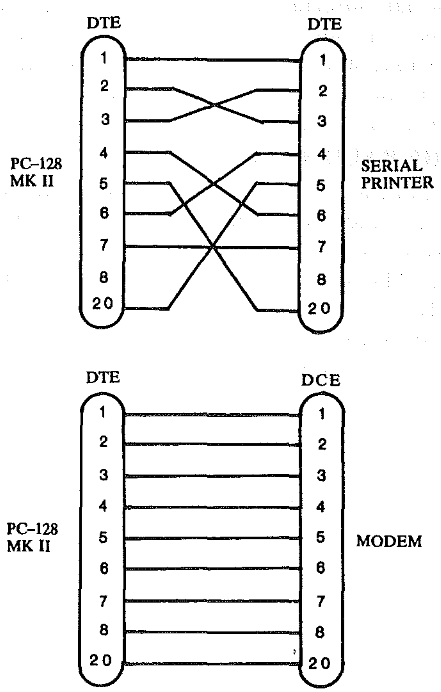


FIG 17–1 (a) Connecting to a serial printer
(b) Connecting to a modem

USING SERIAL PRINTER

The LINGO PC−128 mkII can be connected to serial printer through the RS−232C serial interface. For proper operation, both th serial printer and computer must be set to the same parameters such as baud rate, word length, no. of stop bits and parity.

Set dipswitch SW2 on the system board according to the printer baud rate. The procedures of using serial printer under different operating systems are as follows:−

CP/M 3

1) The serial interface should be configured as in slot 2 (factory setting).

2) Run the program SERIAL.COM in the CP/M 3 Utility disk to setup the word length and no. of stop bits. Answer the questions as prompt by the program.

3) Using DEVICE command to assign the physical device SERIAL to the logical device LST:, i.e.

   A > DEVICE LST:=SERIAL

4) Type CTRL−P to activate printer.

5) To deactivate printer, type CTRL−P again.

CP/M 2.2

1) The serial interface should be configured as in slot 1. Refer to Section 18 under JUMPER SELECTION on Jumper JD and JE for proper configuration.

2) Run the program SERIAL.COM in the CP/M 3 Utility disk. Answer the questions as prompt by the program.

3) Type CTRL−P to activate the printer

DOS 3.3

1) Most application software requires slot 1 to be the printer port. The serial interface should therefore be configured as in slot 1. Refer to Section 18 under JUMPER SELECTION on Jumper JD and JE for proper installation.

2) The default parameters are as follows:–

Baud rate	=	300
Word length	=	8 bits
No. of stop bits	=	2 bits
Parity	=	No parity

3) Type PR#1 to activate the printer and PR#0 to deactivate.

OPERATION MODES AND CONTROL KEYS

The serial interface can be operated in two different modes; Non–terminal and Terminal mode. In the non–terminal mode, the Lingo PC–128 mkII can be connected to peripherals such as printers and terminals. Input to the Lingo PC–128 mkII can be taken from the serial port instead of the keyboard and output from the Lingo PC–128 mkII can be sent to the serial port as well as to the computer's display.

In the Terminal mode, the firmware supports both full– and half–duplex modes and the BREAK signal. The user can switch from a non–terminal mode to the terminal mode with a few commands entered from the computer's keyboard. In the Terminal mode, the Lingo PC–128 mkII will simulate as a dumb terminal. It will send the characters from the keyboard to the remote computer and print the characters which come from the remote computer onto the screen, through the use of a modem.

NON–TERMINAL MODE

OUTPUT

If you are connecting a serial printer, you will need to tell the Lingo PC–128 mkII when to stop sending output to the console and where to send it so that the printer can receive it. To do this, enter the following commands from the keyboard:

From BASIC : PR#2
or
From Monitor : 2^P

All subsequent outputs from the computer will now be handled by the serial interface firmware's output routine. Anything typed at the keyboard will thus go to the printer or other peripheral, and will not appear on the screen. To restore the screen as the output destination, type in the following commands :

From BASIC : PR#0
or
From Monitor : O^P

The output from the computer can also be sent to the serial interface under program control. The following example shows how to activate and de–activate the serial interface:

```
10 D$=CHR$(4)
20 PRINT D$;"PR#2"
30 PRINT "THIS SHOULD BE SENT TO THE SERIAL
PRINTER"
40 PRINT D$;"PR#0"
50 END
```

INPUT

If you are connecting the serial interface to another keyboard, paper tape reader, or other input device to the Lingo PC–128 mkII, you will need to tell the computer when to stop expecting input from the main keyboard and where to go for future input. To do this, type the following commands:

From BASIC : IN#2
or
From Monitor : 2^K
Either of these commands will cause your computer to go to the serial interface firmware's input routines for all subsequent input. The Lingo PC–128 mkII will respond only to the RESET and ^C' entered from the keyboard. Therefore you should enter the PR#2 command before the IN#2 command if you wish to both input and output through the serial interface. To restore the Lingo PC–128 mkII keyboard as the input device, enter IN#0 or 0K from the new input device (if it has a keyboard) or under program control, or enter RESET or '^C' from the Lingo PC–128 mkII keyboard.

TERMINAL MODE

In the Terminal mode, the computer simulates as a computer terminal. The Terminal Commands entered from the keyboard are always prefixed by ^A. They can be used only after an IN#2 command and are the only keyboard commands, besides RESET and^C, that the Lingo PC−128 mkII will recognise after IN#2 has been entered. The various terminal commands are as described below:

^A^F — Full Duplex

The Full−Duplex command allows your Lingo PC−128 mkII to operate as a full−duplex terminal. In the full duplex mode, when a character is entered on the keyboard, it is sent to the external device that the terminal is communicating with, Unless the external device sends it back (echoes), the character does not appear on the screen. The full−duplex mode with echoing is often preferred since it confirms the communication's reliability. However, it requires the receiving device to have the capability of echoing the character. Enter^A^F to enable the full−duplex mode.

^A^H — Half−Duplex

The Half−Duplex command causes your Lingo PC−128 mkII to mimic a half−duplex terminal. In the half−duplex mode, a character entered on the keyboard is sent to both the Lingo PC−128 mkII's screen and the external device. Thus the user sees exactly what he has entered. If you use the half−duplex mode with a receiving device designed for full−duplex, every character on the screen wil be ddoouubblleedd, since the receiving device will be echoing the characters it receives.

^A^S — Break Command

This command allows the Lingo PC−128 mkII to send out the BREAK signal. It is needed only when communicating with an external device that requires the BREAK signal. Turn on the BREAK signal by entering ^A^S. Turn it off by pressing any other key.

^A^X — Exit Commands

To exit from the terminal mode, enter^A^X.A backlash will appear on the screen. If a PR#2 command was in effect, it does not change. Input, however, will now be taken from the Lingo PC−128 mkII keyboard. RESET and^C would also return the Lingo PC−128 mkII to normal operations; however, any program running would be terminated. After RESET, both PR#2 and IN#2 are set to their default values, IN#0 and PR#0.

^R Remote Mode

When the Lingo PC−128 mkII is operating in the terminal mode, an external device can connect and disconnect the Lingo's "brain". If the Lingo PC−128 mkII receives a^R while in the terminal mode, it operates in a Remote Mode to the external device. Any input coming from the external device goes to the Lingo's brain. Thus the external device control the Lingo PC−128 mkII as if its instructions had come from the Lingo PC−128 mkII keyboard. If the PR#2 command is in effect, the Lingo will echo what it receives from the external device.

^T Terminal Mode

After a^R commmand has been issued,^T returns the Lingo PC128 mkII to the terminal mode. In effect, it cancels the Remote Mode command and reinstates the Lingo as a half−duplex or full−duplex terminal. If the terminal mode has not yet been invoked,^T sent by an external device turns the Lingo into a half−duplex terminal.

CHANGING PARAMETERS

Some of the serial firmware parameters can be changed by BASIC "POKE" command or monitor commands. To change the parameters, you must first initialize the serial interface by using IN#2 or PR#2 commands or the monitor equivalents, 2^P or 2^K.Any subsequent invoking of the IN#2 or PR#2 comands or monitor equivalents will reinitialize the serial interface, causing the parameters to return to their default values. These parameters are as follows:

(1) SERIAL DATA FORMAT

The default value for the ACIA (Asynchronous Communication Interface Adapter) operating command is $11, which specifies an 8 bit word, 2 stop bits, no parity, and no interrupts. To change the operating mode, POKE the desired data into location 16288 ($C0A0). If the serial interface has been reconfigured as slot#1 (see Section 18), then POKE the desired data into location 16272 ($C090).

(2) LOWER−CASE CONVERSION

The Lingo PC−128 mkII converts lower−case characters received from the external device into upper−case characters. To stop this, use POKE 1594,0 ($63A). If the serial interface has been configured as slot#1, use POKE 1593,0 ($639) instead. If you load this location with 160 ($A0), lower−characters are printed as upper−case.

(3) OUTPUT TO THE SCREEN

The serial interface will echo output characters to the Lingo PC−128 mkII's screen only if bit 7 at $6FA is '1' (for slot#1 use $6F9). From BASIC, POKE 1786,128 (use POKE 1785,128 for slot#1) to enable screen echo and POKE 1786,0 (use POKE 1785,0 for slot#1) to disable screen echo.

PROGRAMMING INFORMATION

This section will describe the technical details of the ACIA (Asynchronous Communication Interface Adapter), to allow you to write your own assembly−language driver routines for the serial interface.It is intended for assembly programmers and novices who are familiar with assembly language programming.

ACIA REGISTERS

The Lingo PC−128 mkII dedicates 16 memory addresses for each peripheral connector slot.These 16 addresses are above and beyond the 256 dedicated program memory addresses-.The I/O addresses occupied by the ACIA registers are as follows:

ACIA Registers	Slot#1	Slot#2	R/W
Command Register	$C090	$C0A0	Write
Status Register	$C090	$C0A0	Read
Transmit Register	$C091	$C0A1	Write
Receive Register	$C091	$C0A1	Read

Note: The last address digit is not decoded beyond even or odd. This means that data can be passed through any odd address within the range, while the ACIA's command/status registers can be accessed at any even address.

ACIA COMMAND REGISTER

ACIA operation is controlled by one−byte commands written to the Command Register.Because the baud rate generator outputs a clock 16 times the baud rate, bits 1 and 0 should be set to "01". The commands are defined as follows:

D7	D6	D5	D4	D3	D2	D1	D0	ACIA Commands
x	x	x	x	x	x	0	0	The clock is 1x the baud rate.
x	x	x	x	x	x	0	1	The clock is 16x the baud rate.
x	x	x	x	x	x	1	0	The clock is 64x the baud rate.
x	x	x	x	x	x	1	1	ACIA master Reset
x	x	x	0	0	0	x	x	7 data + Even parity + 2 Stop bits.
x	x	x	0	0	1	x	x	7 data + Odd parity + 2 Stop bits.
x	x	x	0	1	0	x	x	7 data + Even parity + 1 Stop bit.
x	x	x	0	1	1	x	x	7 data + Odd parity + 1 Stop bit.
x	x	x	1	0	0	x	x	8 data + No parity+ 4 Stop bits.
x	x	x	1	0	1	x	x	8 data + No parity+ 1 Stop bit.
x	x	x	1	1	0	x	x	8 data + Even parity + 1 Stop bit.
x	x	x	1	1	1	x	x	8 data + Odd parity + 1 Stop bit.
x	0	0	x	x	x	x	x	Set RS−232C CTS Disable transmit interrupts.
x	0	1	x	x	x	x	x	Set RS−232C CTS Enbale transmit interrupts.
x	1	0	x	x	x	x	x	Clear RS−232C CTS Disable transmit interrupts.
x	1	1	x	x	x	x	x	Set RS−232C CTS Transmit break on transmit data Disable transmit interrupts.
0	x	x	x	x	x	x	x	Disable Receive Interrupts.
1	x	x	x	x	x	x	x	Enable Receive Interrupts.

ACIA STATUS REGISTER

The status bits in the Status Register will indicate the status of the ACIA, and are as described below:

Bit	Function	ACIA Status: If "1" indicates
0	RDRF	Receive Data Register Full —data received
1	TDRE	Transmit Data Register Empty —data sent
2	DCD	Data Carrier Detect —carrier from modem not present
3	CTS	Clear to Send —inhibits
4	FE	Framing Error —data not properly received
5	OVRN	Over—Run Error —data received before previous byte read
6	PE	Parity Error
7	IRQ	Interrupt Request —ACIA generated transmit or receive interrupt.

RS–232C CONNECTOR PINOUT

The signal connections for the RS–232C connector are as follows:

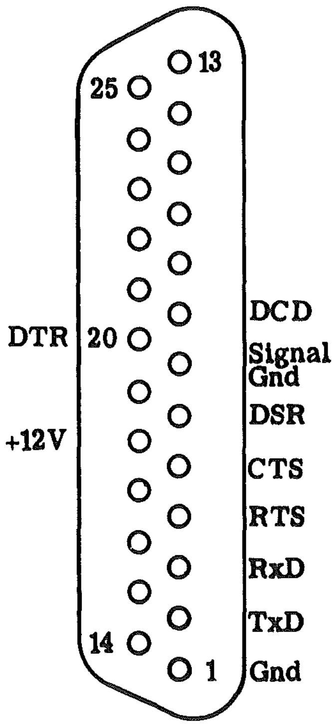


FIG 17–2 Serial Interface Pinout

SECTION 18 JUMPER SELECTION

Contents **Page**

INTRODUCTION

Inside the Lingo PC–128 mkII, there are several terminal pins on the motherboard, whereby "jumpers" are inserted onto them. These "jumpers" are used to electrically short two of the terminal pins, so as to configure a certain function on the motherboard. You can also remove these jumpers and insert them onto the other of the terminal pins so as to configure a different function.

There are altogether six jumpers on the motherboard, and are numbered as JA, JC, JD, JE, and JF. Each of these jumpers serve a different function and are described below:

JUMPER JA

This jumper may be used only in conjunction with the optional Rom–Card. The Lingo PC–128 mkII has a ROM slot in which an optional ROM–Card can be inserted. This ROM–Card can contain one, or two BASIC Interpreters.

Jumper JA has been connected to switch SW4 (is located at the rear of the computer) such that, switching SW4 to the right will connect JA–pin 1 to 0V, while switching SW4 to the left will connect JA–pin 3 to 0V. Also, when SW4 is switched to the right, the 80 column mode is disabled. Jumper JA can be wired such that 2 BASIC interpreters on the optional ROM–card can be selected by setting switch SW4.

Note: SW4 is labelled as '40/80 COLUMN' on the rear panel.

JUMPER JC

Jumper JC allows you to select between black and white (B+W), and Phase–Alternate–Line (PAL) output to the video connector CN4 (labelled as B+W/PAL). When Jumper JC is in position 1–2 (factory setting), the video output at CN4 is B+W. This allows the connection of Monochrome, Green or Amber Monitors to the Lingo PC–128 mkII. When Jumper JC is in position 2–3, the unmodulated PAL signal is available at the video connector CN4. This allows a PAL monitor to be connected to the Lingo PC–128 mkII.

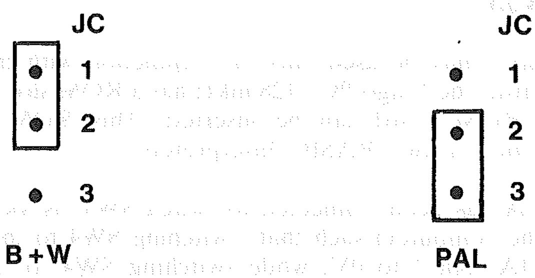


FIG 18–1 Jumper JC

JUMPERS JD AND JE

The on–board printer and serial interfaces, each occupies an I/O slot. The printer interface is in slot 1, while the serial interface is in slot 2 (factory setting). But the serial interface can configured in slot 1, by setting jumpers JD and JE.

The two jumpers at JD routes the "device select" signals $C09x and $C0Ax to the printer and serial interface respectively. Similarly, the two jumpers at JE routes the "I/O select" signals $C1xx and $C2xx to the printer and serial interface respectively. When the serial interface is configured in slot1, the printer interface is disabled.

To configure the serial interface in slot1, perform the following steps:

(1) Remove all jumpers at JD and JE.

(2) Insert jumper across JD pin 2 and pin 4.

(3) Insert jumper across JE pin 2 and pin 4.

FIG 18–2 Parallel slot 1, Serial slot 2

FIG 18–3 Serial slot 1, Parallel disabled

JUMPER JF

The Lingo PC−128 mkII can display in both 40 and 80 column mode. The 80 column display is designed such that it emulates a Videx Videoterm 80 column card. In BASIC, the display can be switched to 80 columns by typing "PR#3". The 80 column mode can be permanently cutoff by removing Jumper JF. In this way, application softwares will assume the absence of the 80 column mode, and will always display in 40 columns only.

FIG 18–4 Jumper JF

APPENDICES

Contents **Page**

APPENDIX A

SPECIAL I/O LOCATIONS

The table below is a summary of the special I/O addresses and its corresponding function:

Address Hex	Decimal	Read	Write
$C000	49152	Keyboard Input	Inverse Display
$C001	49153		Flash Display
$C002	49154		Character Set 0
$C003	49155		Character Set 1
$C008	49160	Keyboard Status	Keyboard Interrupt Off
$C009	49161		Keyboard Interrupt On
$C00A	49162		80 Character Display Off
$C00B	49163		80 Character Display On
$C00C	49164		Dynamic RAM On
$C00D	49165		Dynamic RAM Off
$C00E	49166		$C08x Addresses Active
$C00F	49167		$C08x Addresses Inactive
$C010	49168	Clear Keyboard Strobe	
$C030	49200	Toggle Speaker output	
$C04x	49216	Utility Strobe	Utility Strobe
$C050	49232	Graphic Mode On	
$C051	49233	Text Mode On	
$C052	49234	Full Graphics Mode	
$C053	49235	Mixed Graphics Mode	
$C054	49236	Select Page 1	
$C055	49237	Select Page 2	
$C056	49238	Lo–Res or Med–Res	
$C057	49239	Hi–Res	
$C058	49240	AN0 = '0'	AN0 = '0'
$C059	49241	AN0 = '1'	AN0 = '1'
$C05A	49242	AN1 = '0'	AN1 = '0'
$C05B	49243	AN1 = '1'	AN1 = '1'
$C05C	49244	AN2 = '0'	AN2 = '0'
$C05D	49245	AN2 = '1'	AN2 = '1'
$C05E	49246	AN3 = '0'	AN3 = '0'
$C05F	49247	AN3 = '1'	AN3 = '1'
$C060	49248		Select $0000–$1FFF Bank 0
$C061	49249	Pushbutton PB1	Select $0000–$1FFF Bank 1
$C062	49250	Pushbutton PB2	Select $2000–$3FFF Bank 0
$C063	49251	Pushbutton PB3	Select $2000–$3FFF Bank 1
$C064	49252	GC0	Select $4000–$5FFF Bank 0
$C065	49253	GC1	Select $4000–$5FFF Bank 1
$C066	49254	GC2	Select $6000–$7FFF Bank 0
$C067	49255	GC3	Select $6000–$7FFF Bank 1

Address Hex	Address Decimal	Read	Write
$C068	49256	....	Select $8000–$9FFF Bank 0
$C069	49257	....	Select $8000–$9FFF Bank 1
$C06A	49258	....	Select $A000–$BFFF Bank 0
$C06B	49259	....	Select $A000–$BFFF Bank 1
$C06C	49260	....	Select $D000–$DFFF Bank 0
$C06D	49261	....	Select $D000–$DFFF Bank 1
$C06E	49262	....	Select $E000–$FFFF Bank 0
$C06F	49263	....	Select $E000–$FFFF Bank 1
$C070	49264	Start Analog x Timers	Start Analog x Timers
$C08x	49280	Language Card Control	
$C09x	49296	Parallel Interface	Parallel Interface
$C0Ax	49312	Serial Interface	Serial Interface
$C0Cx	49344	Slot 4 Device Select	Slot 4 Device Select
$C0Dx	49360	Slot 5 Device Select	Slot 5 Device Select
$C0Ex	49376	Slot 6 Device Select	Slot 6 Device Select
$C0Fx	49329	Slot 7 Device Select	Slot 7 Device Select
$C2xx	50688	....	Z–80 On/Off

Standard 7–bit ASCII Codes

The ASCII Character Set

Decimal:		128	144	16Ø	176	192	2Ø8	224	24Ø
	Hex:	$8Ø	$9Ø	$AØ	$BØ	$CØ	$DØ	$EØ	$FØ
Ø	$Ø	nul	dle		Ø	@	P		p
1	$1	soh	dc1	!	1	A	Q	a	q
2	$2	stx	dc2	"	2	B	R	b	r
3	$3	etx	dc3	#	3	C	S	c	s
4	$4	eot	dc4	$	4	D	T	d	t
5	$5	enq	nak	%	5	E	U	e	u
6	$6	ack	syn	&	6	F	V	f	v
7	$7	bel	etb		7	G	W	g	w
8	$8	bs	can	(	8	H	X	h	x
9	$9	ht	em	)	9	I	Y	i	y
1Ø	$A	lf	sub	*	:	J	Z	j	z
11	$B	vt	esc	+	;	K	[	k	{
12	$C	ff	fs	,	<	L	\	l	|
13	$D	cr	gs	–	=	M	]	m	}
14	$E	so	rs	.	>	N	^	n	~
15	$F	si	us	/	?	O	_	o	rub

APPENDIX B

JUMPER PADS

There are several "jumper pads" on the reverse side of the Lingo PC−128 mkII motherboard. These jumper pads are in the form of a "O" and a "X" as shown below. Modifications are made by cutting the "X" or/and joining the "O". In order to make these modifications, the user is expected to have experience in handling a soldering iron.

All the jumper pads can be found on the reverse of the motherboard. The usage of each of these jumper pads are as described below:

PAL to NTSC Modification

The Lingo PC−128 mkII video output can produce PAL signal, by configuring the on−board "jumpers". This is explained in Chapter 7 on "Jumper Selection". But it can also be modified to produce composite NTSC signals so that it can be connected to an NTSC monitor.

In order to modify the existing PAL to NTSC signals, perform the following procedures:

(a) Cut the "X" jumper pads of P1B, P2B, P4B, and P5B.

(b) Solder together the "O" jumper pads of P1A, P2A, P4A, P5A, and P6A.

(c) Replace XTAL1 with a 3.579545MHz crystal.

(d) Replace XTAL3 with a 14.31818MHz crystal.

Slot 7 to Slot 3 Conversion

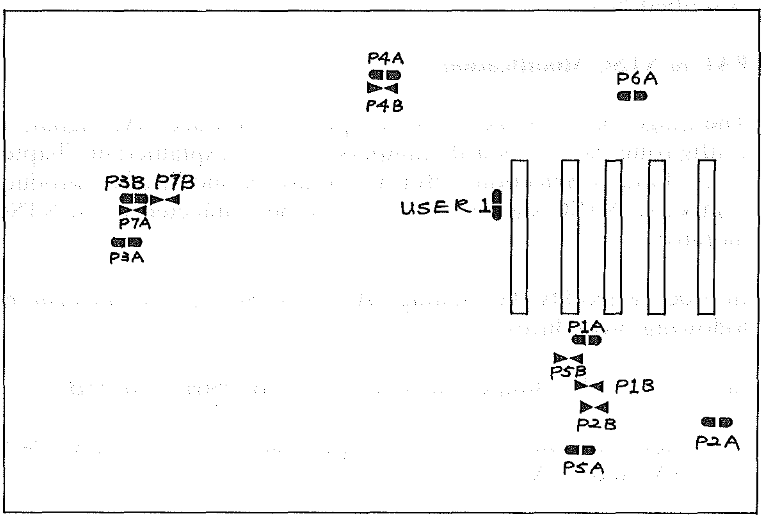
The peripheral I/O slot 7 can be modified as slot 3. This should be done only when slot 3 is needed for use with other display cards, instead of the on−board 80 column display. This modification procedures are as follows :

(a) Cut the "X" jumper pads of P7A and P7B.

(b) Solder together the "O" jumper pads of P3A and P3B.

USER 1 JUMPER

The location of the 'USER l' jumper pad on the reverse side of the motherboard, is as shown below. If this "O" jumper pad is soldered, then the USER 1 line on the I/O peripheral connectors becomes active (Pin 39). If any peripheral card pulls this line low, all on–board I/O decoding is disabled. The I/O SELECT and the DEVICE SELECT lines all go high and will remain while USER 1 is low, regardless of the address on the address bus.



Solder Side of Motherboard

APPENDIX C

Z–80 INSTRUCTION SET

This Appendix contains the Z–80 instruction set.

SUMMARY OF FLAG OPERATION

Instruction	D7 S	Z		H		P/V	N	D0 C	Comments
ADD A,s; ADC A,s	↕	↕	X	↕	X	V	0	↕	8-bit add or add with carry
SUB s; SBC A,s; CP s; NEG	↕	↕	X	↕	X	V	1	↕	8-bit subtract, subtract with carry, compare and negate accumulator
AND s	↕	↕	X	1	X	P	0	0	Logical operations
OR s; XOR s	↕	↕	X	0	X	P	0	0	Logical operations
INC s	↕	↕	X	↕	X	V	0	•	8-bit increment
DEC s	↕	↕	X	↕	X	V	1	•	8-bit decrement
ADD DD, SS	•	•	X	X	X	•	0	↕	16-bit add
ADC HL, SS	↕	↕	X	X	X	V	0	↕	16-bit add with carry
SBC HL, SS	↕	↕	X	X	X	V	1	↕	16-bit subtract with carry
RLA; RLCA; RRA; RRCA	•	•	X	0	X	•	0	↕	Rotate accumulator
RL s; RLC s; RR s; RRC s; SLA s; SRA s; SRL s	↕	↕	X	0	X	P	0	↕	Rotate and shift locations
RLD; RRD	↕	↕	X	0	X	P	0	•	Rotate digit left and right
DAA	↕	↕	X	↕	X	P	•	↕	Decimal adjust accumulator
CPL	•	•	X	1	X	•	1	•	Complement accumulator
SCF	•	•	X	0	X	•	0	1	Set carry
CCF	•	•	X	X	X	•	0	↕	Complement carry
IN r, (C)	↕	↕	X	0	X	P	0	•	Input register indirect
INI; IND; OUTI; OUTD	X	↕	X	X	X	X	1	X	Block input and output
INIR; INDR; OTIR; OTDR	X	1	X	X	X	X	1	X	Z = 0 if B ≠ 0 otherwise Z = 1
LDI; LDD	X	X	X	0	X	↕	0	•	Block transfer instructions
LDIR; LDDR	X	X	X	0	X	0	0	•	P/V = 1 if BC ≠ 0, otherwise P/V = 0
CPI; CPIR; CPD; CPDR	↕	↕	X	↕	X	↕	1	•	Block search instructions Z = 1 if A = (HL), otherwise Z = 0 P/V = 1 if BC ≠ 0, otherwise P/V = 0
LD A, I; LD A, R	↕	↕	X	0	X	IFF	0	•	The content of the interrupt enable flip-flop (IFF) is copied into the P/V flag
BIT b, s	X	↕	X	1	X	X	0	•	The state of bit b of location s is copied into the Z flag

The following notation is used in this table:

SYMBOL	OPERATION
C	Carry/link flag. C=1 if the operation produced a carry from the MSB of the operand or result.
Z	Zero flag. Z=1 if the result of the operation is zero.
S	Sign flag. S=1 if the MSB of the result is one.
P/V	Parity or overflow flag. Parity (P) and overflow (V) share the same flag. Logical operations affect this flag with the parity of the result while arithmetic operations affect this flag with the overflow of the result. If P/V holds parity, P/V=1 if the result of the operation is even, P/V=0 if result is odd. If P/V holds overflow, P/V=1 if the result of the operation produced an overflow.
H	Half-carry flag. H=1 if the add or subtract operation produced a carry into or borrow from bit 4 of the accumulator.
N	Add/Subtract flag. N=1 if the previous operation was a subtract. H and N flags are used in conjunction with the decimal adjust instruction (DAA) to properly correct the result into packed BCD format following addition or subtraction using operands with packed BCD format.
	The flag is affected according to the result of the operation.
•	The flag is unchanged by the operation.
0	The flag is reset by the operation.
1	The flag is set by the operation.
X	The flag is a "don't care".
V	P/V flag affected according to the overflow result of the operation.
P	P/V flag affected according to the parity result of the operation.
r	Any one of the CPU registers A, B, C, D, E, H, L.
s	Any 8-bit location for all the addressing modes allowed for the particular instruction.
ss	Any 16-bit location for all the addressing modes allowed for that instruction.
ii	Any one of the two index registers IX or IY.
R	Refresh counter.
n	8-bit value in range <0, 255>
nn	16-bit value in range <0, 65535>

8-BIT LOAD GROUP

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
LD r, s	r ← s	•	•	X	•	X	•	•	•	01 r s		1	1	4	r, s Reg.
LD r, n	r ← n	•	•	X	•	X	•	•	•	00 r 110		2	2	7	000 B
										← n →					001 C
LD r, (HL)	r ← (HL)	•	•	X	•	X	•	•	•	01 r 110		1	2	7	010 D
LD r, (IX+d)	r ← (IX+d)	•	•	X	•	X	•	•	•	11 011 101	DD	3	5	19	011 E
										01 r 110					100 H
										← d →					101 L
LD r, (IY+d)	r ← (IY+d)	•	•	X	•	X	•	•	•	11 111 101	FD	3	5	19	111 A
										01 r 110					
										← d →					
LD (HL), r	(HL) ← r	•	•	X	•	X	•	•	•	01 110 r		1	2	7	
LD (IX+d), r	(IX+d) ← r	•	•	X	•	X	•	•	•	11 011 101	DD	3	5	19	
										01 110 r					
										← d →					
LD (IY+d), r	(IY+d) ← r	•	•	X	•	X	•	•	•	11 111 101	FD	3	5	19	
										01 110 r					
										← d →					
LD (HL), n	(HL) ← n	•	•	X	•	X	•	•	•	00 110 110	36	2	3	10	
										← n →					
LD (IX+d), n	(IX+d) ← n	•	•	X	•	X	•	•	•	11 011 101	DD	4	5	19	
										00 110 110	36				
										← d →					
										← n →					
LD (IY+d), n	(IY+d) ← n	•	•	X	•	X	•	•	•	11 111 101	FD	4	5	19	
										00 110 110	36				
										← d →					
										← n →					
LD A, (BC)	A ← (BC)	•	•	X	•	X	•	•	•	00 001 010	0A	1	2	7	
LD A, (DE)	A ← (DE)	•	•	X	•	X	•	•	•	00 011 010	1A	1	2	7	
LD A, (nn)	A ← (nn)	•	•	X	•	X	•	•	•	00 111 010	3A	3	4	13	
										← n →					
										← n →					
LD (BC), A	(BC) ← A	•	•	X	•	X	•	•	•	00 000 010	02	1	2	7	
LD (DE), A	(DE) ← A	•	•	X	•	X	•	•	•	00 010 010	12	1	2	7	
LD (nn), A	(nn) ← A	•	•	X	•	X	•	•	•	00 110 010	32	3	4	13	
										← n →					
										← n →					
LD A, I	A ← I	↕	↕	X	0	X	IFF	0	•	11 101 101	ED	2	2	9	
										01 010 111	57				
LD A, R	A ← R	↕	↕	X	0	X	IFF	0	•	11 101 101	ED	2	2	9	
										01 011 111	5F				
LD I, A	I ← A	•	•	X	•	X	•	•	•	11 101 101	ED	2	2	9	
										01 000 111	47				
LD R, A	R ← A	•	•	X	•	X	•	•	•	11 101 101	ED	2	2	9	
										01 001 111	4F				

Notes: r, s means any of the registers A, B, C, D, E, H, L
IFF the content of the interrupt enable flip-flop (IFF) is copied into the P/V flag

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown,
↕ = flag is affected according to the result of the operation.

16-BIT LOAD GROUP

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
LD dd, nn	$dd \leftarrow nn$	•	•	X	•	X	•	•	•	00 dd0 001 ← n → ← n →		3	3	10	dd Pair 00 BC 01 DE 10 HL 11 SP
LD IX, nn	$IX \leftarrow nn$	•	•	X	•	X	•	•	•	11 011 101 00 100 001 ← n → ← n →	DD 21	4	4	14	
LD IY, nn	$IY \leftarrow nn$	•	•	X	•	X	•	•	•	11 111 101 00 100 001 ← n → ← n →	FD 21	4	4	14	
LD HL, (nn)	$H \leftarrow (nn+1)$ $L \leftarrow (nn)$	•	•	X	•	X	•	•	•	00 101 010 ← n → ← n →	2A	3	5	16	
LD dd, (nn)	$dd_H \leftarrow (nn+1)$ $dd_L \leftarrow (nn)$	•	•	X	•	X	•	•	•	11 101 101 01 dd1 011 ← n → ← n →	ED	4	6	20	
LD IX, (nn)	$IX_H \leftarrow (nn+1)$ $IX_L \leftarrow (nn)$	•	•	X	•	X	•	•	•	11 011 101 00 101 010 ← n → ← n →	DD 2A	4	6	20	
LD IY, (nn)	$IY_H \leftarrow (nn+1)$ $IY_L \leftarrow (nn)$	•	•	X	•	X	•	•	•	11 111 101 00 101 010 ← n → ← n →	FD 2A	4	6	20	
LD (nn), HL	$(nn+1) \leftarrow H$ $(nn) \leftarrow L$	•	•	X	•	X	•	•	•	00 100 010 ← n → ← n →	22	3	5	16	
LD (nn), dd	$(nn+1) \leftarrow dd_H$ $(nn) \leftarrow dd_L$	•	•	X	•	X	•	•	•	11 101 101 01 dd0 011 ← n → ← n →	ED	4	6	20	
LD (nn), IX	$(nn+1) \leftarrow IX_H$ $(nn) \leftarrow IX_L$	•	•	X	•	X	•	•	•	11 011 101 00 100 010 ← n → ← n →	DD 22	4	6	20	
LD (nn), IY	$(nn+1) \leftarrow IY_H$ $(nn) \leftarrow IY_L$	•	•	X	•	X	•	•	•	11 111 101 00 100 010 ← n → ← n →	FD 22	4	6	20	
LD SP, HL	$SP \leftarrow HL$	•	•	X	•	X	•	•	•	11 111 001	F9	1	1	6	
LD SP, IX	$SP \leftarrow IX$	•	•	X	•	X	•	•	•	11 011 101 11 111 001	DD F9	2	2	10	
LD SP, IY	$SP \leftarrow IY$	•	•	X	•	X	•	•	•	11 111 101 11 111 001	FD F9	2	2	10	
PUSH qq	$(SP-2) \leftarrow qq_L$ $(SP-1) \leftarrow qq_H$	•	•	X	•	X	•	•	•	11 qq0 101		1	3	11	qq Pair 00 BC 01 DE 10 HL 11 AF
PUSH IX	$(SP-2) \leftarrow IX_L$ $(SP-1) \leftarrow IX_H$	•	•	X	•	X	•	•	•	11 011 101 11 100 101	DD E5	2	4	15	
PUSH IY	$(SP-2) \leftarrow IY_L$ $(SP-1) \leftarrow IY_H$	•	•	X	•	X	•	•	•	11 111 101 11 100 101	FD E5	2	4	15	
POP qq	$qq_H \leftarrow (SP+1)$ $qq_L \leftarrow (SP)$	•	•	X	•	X	•	•	•	11 qq0 001		1	3	10	
POP IX	$IX_H \leftarrow (SP+1)$ $IX_L \leftarrow (SP)$	•	•	X	•	X	•	•	•	11 011 101 11 100 001	DD E1	2	4	14	
PUP IY	$IY_H \leftarrow (SP+1)$ $IY_L \leftarrow (SP)$	•	•	X	•	X	•	•	•	11 111 101 11 100 001	FD E1	2	4	14	

Notes: dd is any of the register pairs BC, DE, HL, SP
qq is any of the register pairs AF, BC, DE, HL
$(PAIR)_H$, $(PAIR)_L$ refer to high order and low order eight bits of the register pair respectively.
e.g. $BC_L = C$, $AF_H = A$

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown,
↕ flag is affected according to the result of the operation.

EXCHANGE GROUP AND BLOCK TRANSFER AND SEARCH GROUP

Mnemonic	Symbolic Operation	Flags S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
EX DE, HL	DE↔HL	•	•	X	•	X	•	•	•	11 101 011	EB	1	1	4	
EX AF, AF'	AF↔AF'	•	•	X	•	X	•	•	•	00 001 000	08	1	1	4	
EXX	(BC↔BC' DE↔DE' HL↔HL')	•	•	X	•	X	•	•	•	11 011 001	D9	1	1	4	Register bank and auxiliary register bank exchange
EX (SP), HL	H ↔(SP+1) L ↔(SP)	•	•	X	•	X	•	•	•	11 100 011	E3	1	5	19	
EX (SP), IX	IX_H↔(SP+1) IX_L↔(SP)	•	•	X	•	X	•	•	•	11 011 101 11 100 011	DD E3	2	6	23	
EX (SP), IY	IY_H↔(SP+1) IY_L↔(SP)	•	•	X	•	X	•	•	•	11 111 101 11 100 011	FD E3	2	6	23	
LDI	(DE)←(HL) DE ← DE+1 HL ← HL+1 BC ← BC-1	•	•	X	0	X	① ↕	0	•	11 101 101 10 100 000	ED A0	2	4	16	Load (HL) into (DE), increment the pointers and decrement the byte counter (BC)
LDIR	(DE)←(HL) DE ← DE+1 HL ← HL+1 BC ← BC-1 Repeat until BC = 0	•	•	X	0	X	0	0	•	11 101 101 10 110 000	ED B0	2 2	5 4	21 16	If BC ≠ 0 If BC = 0
LDD	(DE)←(HL) DE ← DE-1 HL ← HL-1 BC ← BC-1	•	•	X	0	X	① ↕	0	•	11 101 101 10 101 000	ED A8	2	4	16	
LDDR	(DE)←(HL) DE ← DE-1 HL ← HL-1 BC ← BC-1 Repeat until BC = 0	•	•	X	0	X	0	0	•	11 101 101 10 111 000	ED B8	2 2	5 4	21 16	If BC ≠ 0 If BC = 0
CPI	A – (HL) HL ← HL+1 BC ← BC-1	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 100 001	ED A1	2	4	16	
CPIR	A – (HL) HL ← HL+1 BC ← BC-1 Repeat until A = (HL) or BC = 0	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 110 001	ED B1	2 2	5 4	21 16	If BC ≠ 0 and A ≠ (HL) If BC = 0 or A = (HL)
CPD	A – (HL) HL ← HL-1 BC ← BC-1	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 101 001	ED A9	2	4	16	
CPDR	A – (HL) HL ← HL-1 BC ← BC-1 Repeat until A = (HL) or BC = 0	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 111 001	ED B9	2 2	5 4	21 16	If BC ≠ 0 and A ≠ (HL) If BC = 0 or A = (HL)

Notes: ① P/V flag is 0 if the result of BC-1 = 0, otherwise P/V = 1
② Z flag is 1 if A = (HL), otherwise Z = 0.

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown,
↕ = flag is affected according to the result of the operation.

8-BIT ARITHMETIC AND LOGICAL GROUP

	Symbolic	Flags								Op-Code		No. of	No. of M	No. of T	
Mnemonic	Operation	S	Z		H		P/V	N	C	76 543 210	Hex	Bytes	Cycles	States	Comments
ADD A, r	A ← A + r	↕	↕	X	↕	X	V	0	↕	10 $\boxed{000}$ r		1	1	4	r Reg.
ADD A, n	A ← A + n	↕	↕	X	↕	X	V	0	↕	11 $\boxed{000}$ 110		2	2	7	000 B
										← n →					001 C
															010 D
ADD A, (HL)	A ← A+(HL)	↕	↕	X	↕	X	V	0	↕	10 $\boxed{000}$ 110		1	2	7	011 E
ADD A, (IX+d)	A←A+(IX+d)	↕	↕	X	↕	X	V	0	↕	11 011 101	DD	3	5	19	100 H
										10 $\boxed{000}$ 110					101 L
										← d →					111 A
ADD A, (IY+d)	A←A+(IY+d)	↕	↕	X	↕	X	V	0	↕	11 111 101	FD	3	5	19	
										10 $\boxed{000}$ 110					
										← d →					
ADC A, s	A←A+s+CY	↕	↕	X	↕	X	V	0	↕	$\boxed{001}$					s is any of r, n,
SUB s	A←A - s	↕	↕	X	↕	X	V	1	↕	$\boxed{010}$					(HL), (IX+d),
SBC A, s	A←A - s - CY	↕	↕	X	↕	X	V	1	↕	$\boxed{011}$					(IY+d) as shown for
AND s	A←A ∧ s	↕	↕	X	1	X	P	0	0	$\boxed{100}$					ADD instruction.
OR s	A←A ∨ s	↕	↕	X	0	X	P	0	0	$\boxed{110}$					The indicated bits
XOR s	A←A ⊕ s	↕	↕	X	0	X	P	0	0	$\boxed{101}$					replace the $\boxed{000}$ in
CP s	A - s	↕	↕	X	↕	X	V	1	↕	$\boxed{111}$					the ADD set above.
INC r	r ← r + 1	↕	↕	X	↕	X	V	0	•	00 r $\boxed{100}$		1	1	4	
INC (HL)	(HL)←(HL)+1	↕	↕	X	↕	X	V	0	•	00 110 $\boxed{100}$		1	3	11	
INC (IX+d)	(IX+d) ←	↕	↕	X	↕	X	V	0	•	11 011 101	DD	3	6	23	
	(IX+d)+1									00 110 $\boxed{100}$					
										← d →					
INC (IY+d)	(IY+d) ←	↕	↕	X	↕	X	V	0	•	11 111 101	FD	3	6	23	
	(IY+d)+1									00 110 $\boxed{100}$					
										← d →					
DEC s	s ← s - 1	↕	↕	X	↕	X	V	1	•	$\boxed{101}$					s is any of r, (HL),
															(IX+d), (IY+d) as
															shown for INC.
															DEC same format
															and states as INC.
															Replace $\boxed{100}$ with
															$\boxed{101}$ in OP Code.

Notes: The V symbol in the P/V flag column indicates that the P/V flag contains the overflow of the result of the operation. Similarly the P symbol indicates parity. V = 1 means overflow, V = 0 means not overflow, P = 1 means parity of the result is even, P = 0 means parity of the result is odd.

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown.
↕ = flag is affected according to the result of the operation.

GENERAL PURPOSE ARITHMETIC AND CPU CONTROL GROUPS

Mnemonic	Symbolic Operation	Flags S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
DAA	Converts acc. content into packed BCD following add or subtract with packed BCD operands	↕	↕	X	↕	X	P	•	↕	00 100 111	27	1	1	4	Decimal adjust accumulator
CPL	$A \leftarrow \overline{A}$	•	•	X	1	X	•	1	•	00 101 111	2F	1	1	4	Complement accumulator (One's complement)
NEG	$A \leftarrow \overline{A}+1$	↕	↕	X	↕	X	V	1	↕	11 101 101 01 000 100	ED 44	2	2	8	Negate acc. (two's complement)
CCF	$CY \leftarrow \overline{CY}$	•	•	X	X	X	•	0	↕	00 111 111	3F	1	1	4	Complement carry flag
SCF	$CY \leftarrow 1$	•	•	X	0	X	•	0	1	00 110 111	37	1	1	4	Set carry flag
NOP	No operation	•	•	X	•	X	•	•	•	00 000 000	00	1	1	4	
HALT	CPU halted	•	•	X	•	X	•	•	•	01 110 110	76	1	1	4	
DI*	$IFF \leftarrow 0$	•	•	X	•	X	•	•	•	11 110 011	F3	1	1	4	
EI*	$IFF \leftarrow 1$	•	•	X	•	X	•	•	•	11 111 011	FB	1	1	4	
IM 0	Set interrupt mode 0	•	•	X	•	X	•	•	•	11 101 101 01 000 110	ED 46	2	2	8	
IM 1	Set interrupt mode 1	•	•	X	•	X	•	•	•	11 101 101 01 010 110	ED 56	2	2	8	
IM 2	Set interrupt mode 2	•	•	X	•	X	•	•	•	11 101 101 01 011 110	ED 5E	2	2	8	

Notes: IFF indicates the interrupt enable flip-flop
CY indicates the carry flip-flop.

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown,
↕ = flag is affected according to the result of the operation.

*Interrupts are not sampled at the end of EI or DI

16-BIT ARITHMETIC GROUP

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments	
ADD HL, ss	HL ← HL+ss	•	•	X	X	X	•	0	↕	00 ss1 001		1	3	11	ss	Reg.
															00	BC
ADC HL, ss	HL ← HL+ss+CY	↕	↕	X	X	X	V	0	↕	11 101 101	ED	2	4	15	01	DE
										01 ss1 010					10	HL
															11	SP
SBC HL, ss	HL ← HL-ss-CY	↕	↕	X	X	X	V	1	↕	11 101 101	ED	2	4	15		
										01 ss0 010						
ADD IX, pp	IX ← IX + pp	•	•	X	X	X	•	0	↕	11 011 101	DD	2	4	15	pp	Reg.
										00 pp1 001					00	BC
															01	DE
															10	IX
															11	SP
ADD IY, rr	IY ← IY + rr	•	•	X	X	X	•	0	↕	11 111 101	FD	2	4	15	rr	Reg.
										00 rr1 001					00	BC
															01	DE
															10	IY
															11	SP
INC ss	ss ← ss + 1	•	•	X	•	X	•	•	•	00 ss0 011		1	1	6		
INC IX	IX ← IX + 1	•	•	X	•	X	•	•	•	11 011 101	DD	2	2	10		
										00 100 011	23					
INC IY	IY ← IY + 1	•	•	X	•	X	•	•	•	11 111 101	FD	2	2	10		
										00 100 011	23					
DEC ss	ss ← ss - 1	•	•	X	•	X	•	•	•	00 ss1 011		1	1	6		
DEC IX	IX ← IX - 1	•	•	X	•	X	•	•	•	11 011 101	DD	2	2	10		
										00 101 011	2B					
DEC IY	IY ← IY - 1	•	•	X	•	X	•	•	•	11 111 101	FD	2	2	10		
										00 101 011	2B					

Notes: ss is any of the register pairs BC, DE, HL, SP
pp is any of the register pairs BC, DE, IX, SP
rr is any of the register pairs BC, DE, IY, SP.

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown.
↕ = flag is affected according to the result of the operation.

ROTATE AND SHIFT GROUP

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
RLCA	CY ← 7 ← 0 A	•	•	X	0	X	•	0	↕	00 000 111	07	1	1	4	Rotate left circular accumulator
RLA	CY ← 7 ← 0 A	•	•	X	0	X	•	0	↕	00 010 111	17	1	1	4	Rotate left accumulator
RRCA	7 → 0 → CY A	•	•	X	0	X	•	0	↕	00 001 111	0F	1	1	4	Rotate right circular accumulator
RRA	7 → 0 → CY A	•	•	X	0	X	•	0	↕	00 011 111	1F	1	1	4	Rotate right accumulator
RLC r		↕	↕	X	0	X	P	0	↕	11 001 011 00 000 r	CB	2	2	8	Rotate left circular register r
RLC (HL)		↕	↕	X	0	X	P	0	↕	11 001 011 00 000 110	CB	2	4	15	r Reg. 000 B 001 C 010 D 011 E 100 H 101 L 111 A
RLC (IX+d)	CY ← 7 ← 0 r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	11 011 101 11 001 011 ← d → 00 000 110	DD CB	4	6	23	
RLC (IY+d)		↕	↕	X	0	X	P	0	↕	11 111 101 11 001 011 ← d → 00 000 110	FD CB	4	6	23	
RL s	CY ← 7 ← 0 s ≡ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	010					Instruction format and states are as shown for RLC's. To form new Op-Code replace 000 of RLC's with shown code
RRC s	7 → 0 → CY s ≡ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	001					
RR s	7 → 0 → CY s ≡ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	011					
SLA s	CY ← 7 ← 0 ← 0 s ≡ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	100					
SRA s	7 → 0 → CY s ≡ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	101					
SRL s	0 → 7 → 0 → CY s ≡ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	111					
RLD	A 7-4 3-0 7-4 3-0 (HL)	↕	↕	X	0	X	P	0	•	11 101 101 01 101 111	ED 6F	2	5	18	Rotate digit left and right between the accumulator and location (HL).
RRD	A 7-4 3-0 7-4 3-0 (HL)	↕	↕	X	0	X	P	0	•	11 101 101 01 100 111	ED 67	2	5	18	The content of the upper half of the accumulator is unaffected

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown,
↕ = flag is affected according to the result of the operation.

BIT SET, RESET AND TEST GROUP

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
BIT b, r	$Z \leftarrow \overline{r_b}$	X	$\updownarrow$	X	1	X	X	0	•	11 001 011	CB	2	2	8	r — Reg.
										01 b r					000 — B
BIT b, (HL)	$Z \leftarrow \overline{(HL)_b}$	X	$\updownarrow$	X	1	X	X	0	•	11 001 011	CB	2	3	12	001 — C
										01 b 110					010 — D
BIT b, $(IX+d)_b$	$Z \leftarrow \overline{(IX+d)_b}$	X	$\updownarrow$	X	1	X	X	0	•	11 011 101	DD	4	5	20	011 — E
										11 001 011	CB				100 — H
										← d →					101 — L
										01 b 110					111 — A
															b — Bit Tested
BIT b, $(IY+d)_b$	$Z \leftarrow \overline{(IY+d)_b}$	X	$\updownarrow$	X	1	X	X	0	•	11 111 101	FD	4	5	20	000 — 0
										11 001 011	CB				001 — 1
										← d →					010 — 2
										01 b 110					011 — 3
															100 — 4
															101 — 5
															110 — 6
															111 — 7
SET b, r	$r_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 001 011	CB	2	2	8	
										$\boxed{11}$ b r					
SET b, (HL)	$(HL)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 001 011	CB	2	4	15	
										$\boxed{11}$ b 110					
SET b, (IX+d)	$(IX+d)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 011 101	DD	4	6	23	
										11 001 011	CB				
										← d →					
										$\boxed{11}$ b 110					
SET b, (IY+d)	$(IY+d)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 111 101	FD	4	6	23	
										11 001 011	CB				
										← d →					
										$\boxed{11}$ b 110					
RES b, s	$s_b \leftarrow 0$ $s \equiv r$, (HL), (IX+d), (IY+d)	•	•	X	•	X	•	•	•	$\boxed{10}$					To form new Op-Code replace $\boxed{11}$ of SET b, s with $\boxed{10}$. Flags and time states for SET instruction

Notes: The notation s_b indicates bit b (0 to 7) or location s.

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown,
$\updownarrow$ = flag is affected according to the result of the operation.

JUMP GROUP

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
JP nn	PC ← nn	•	•	X	•	X	•	•	•	11 000 011	C3	3	3	10	
										← n →					
										← n →					cc Condition
JP cc, nn	If condition cc	•	•	X	•	X	•	•	•	11 cc 010		3	3	10	000 NZ non zero
	is true PC ← nn,									← n →					001 Z zero
	otherwise									← n →					010 NC non carry
	continue														011 C carry
															100 PO parity odd
															101 PE parity even
															110 P sign positive
JR e	PC ← PC + e	•	•	X	•	X	•	•	•	00 011 000	18	2	3	12	111 M sign negative
										← e-2 →					
JR C, e	If C = 0,	•	•	X	•	X	•	•	•	00 111 000	38	2	2	7	If condition not met
	continue									← e-2 →					
	If C = 1,											2	3	12	If condition is met
	PC ← PC+e														
JR NC, e	If C = 1,	•	•	X	•	X	•	•	•	00 110 000	30	2	2	7	If condition not met
	continue									← e-2 →					
	If C = 0,											2	3	12	If condition is met
	PC ← PC+e														
JR Z, e	If Z = 0	•	•	X	•	X	•	•	•	00 101 000	28	2	2	7	If condition not met
	continue									← e-2 →					
	If Z = 1,											2	3	12	If condition is met
	PC ← PC+e														
JR NZ, e	If Z = 1,	•	•	X	•	X	•	•	•	00 100 000	20	2	2	7	If condition not met
	continue									← e-2 →					
	If Z = 0,											2	3	12	If condition is met
	PC ← PC+e														
JP (HL)	PC ← HL	•	•	X	•	X	•	•	•	11 101 001	E9	1	1	4	
JP (IX)	PC ← IX	•	•	X	•	X	•	•	•	11 011 101	DD	2	2	8	
										11 101 001	E9				
JP (IY)	PC ← IY	•	•	X	•	X	•	•	•	11 111 101	FD	2	2	8	
										11 101 001	E9				
DJNZ, e	B ← B-1	•	•	X	•	X	•	•	•	00 010 000	10	2	2	8	If B = 0
	If B = 0,									← e-2 →					
	continue														
	If B ≠ 0,											2	3	13	If B ≠ 0
	PC ← PC+e														

Notes: e represents the extension in the relative addressing mode.
e is a signed two's complement number in the range <126, 129>
e-2 in the op-code provides an effective address of pc+e as PC is incremented by 2 prior to the addition of e.

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown,
↕ = flag is affected according to the result of the operation.

CALL AND RETURN GROUP

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
CALL nn	(SP-1) ← PC_H	•	•	X	•	X	•	•	•	11 001 101	CD	3	5	17	
	(SP-2) ← PC_L									← n →					
	PC ← nn									← n →					
CALL cc, nn	If condition	•	•	X	•	X	•	•	•	11 cc 100		3	3	10	If cc is false
	cc is false									← n →					
	continue,									← n →		3	5	17	If cc is true
	otherwise														
	same as														
	CALL nn														
RET	PC_L ← (SP)	•	•	X	•	X	•	•	•	11 001 001	C9	1	3	10	
	PC_H ← (SP+1)														
RET cc	If condition	•	•	X	•	X	•	•	•	11 cc 000		1	1	5	If cc is false
	cc is false														
	continue,											1	3	11	If cc is true
	otherwise														
	same as														
	RET														
RETI	Return from	•	•	X	•	X	•	•	•	11 101 101	ED	2	4	14	
	interrupt									01 001 101	4D				
RETN[1]	Return from	•	•	X	•	X	•	•	•	11 101 101	ED	2	4	14	
	non maskable									01 000 101	45				
	interrupt														
RST p	(SP-1) ← PC_H	•	•	X	•	X	•	•	•	11 t 111		1	3	11	
	(SP-2) ← PC_L														
	PC_H ← 0														
	PC_L ← p														

cc	Condition	
000	NZ	non zero
001	Z	zero
010	NC	non carry
011	C	carry
100	PO	parity odd
101	PE	parity even
110	P	sign positive
111	M	sign negative

t	p
000	00H
001	08H
010	10H
011	18H
100	20H
101	28H
110	30H
111	38H

[1] RETN loads IFF_2 → IFF_1

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown,
↕ = flag is affected according to the result of the operation.

INPUT AND OUTPUT GROUP

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
IN A, (n)	A ← (n)	•	•	X	•	X	•	•	•	11 011 011 ← n →	DB	2	3	11	n to $A_0 \sim A_7$ Acc to $A_8 \sim A_{15}$
IN r, (C)	r ← (C) if r = 110 only the flags will be affected	↕	↕	X	↕	X	P	0	•	11 101 101 01 r 000	ED	2	3	12	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
INI	(HL) ← (C) B ← B - 1 HL ← HL + 1	X	① ↕	X	X	X	X	1	X	11 101 101 10 100 010	ED A2	2	4	16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
INIR	(HL) ← (C) B ← B - 1 HL ← HL + 1 Repeat until B = 0	X	1	X	X	X	X	1	X	11 101 101 10 110 010	ED B2	2 2	5 (If B ≠ 0) 4 (If B = 0)	21 16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
IND	(HL) ← (C) B ← B - 1 HL ← HL - 1	X	① ↕	X	X	X	X	1	X	11 101 101 10 101 010	ED AA	2	4	16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
INDR	(HL) ← (C) B ← B - 1 HL ← HL - 1 Repeat until B = 0	X	1	X	X	X	X	1	X	11 101 101 10 111 010	ED BA	2 2	5 (If B ≠ 0) 4 (If B = 0)	21 16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
OUT (n), A	(n) ← A	•	•	X	•	X	•	•	•	11 010 011	D3	2	3	11	n to $A_0 \sim A_7$ Acc to $A_8 \sim A_{15}$
OUT (C), r	(C) ← r	•	•	X	•	X	•	•	•	11 101 101 01 r 001	ED	2	3	12	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
OUTI	(C) ← (HL) B ← B - 1 HL ← HL + 1	X	① ↕	X	X	X	X	1	X	11 101 101 10 100 011	ED A3	2	4	16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
OTIR	(C) ← (HL) B ← B - 1 HL ← HL + 1 Repeat until B = 0	X	1	X	X	X	X	1	X	11 101 101 10 110 011	ED B3	2 2	5 (If B ≠ 0) 4 (If B = 0)	21 16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
OUTD	(C) ← (HL) B ← B - 1 HL ← HL - 1	X	① ↕	X	X	X	X	1	X	11 101 101 10 101 011	ED AB	2	4	16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
OTDR	(C) ← (HL) B ← B - 1 HL ← HL - 1 Repeat until B = 0	X	1	X	X	X	X	1	X	11 101 101 10 111 011	ED BB	2 2	5 (If B ≠ 0) 4 (If B = 0)	21 16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$

Notes: ① If the result of B - 1 is zero the Z flag is set, otherwise it is reset.

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown,
↕ = flag is affected according to the result of the operation.

APPENDIX D

6502 INSTRUCTION SET

This Appendix contains the 6502 instruction set.

THE FOLLOWING NOTATION APPLIES TO THIS SUMMARY:

A	Accumulator
X, Y	Index Registers
M	Memory
$\bar{C}$	Borrow
P	Processor Status Register
S	Stack Pointer
✓	Change
—	No Change
+	Add
∧	Logical AND
-	Subtract
⊻	Logical Exclusive Or
↑	Transfer From Stack
↓	Transfer To Stack
→	Transfer To
←	Transfer To
V	Logical OR
PC	Program Counter
PCH	Program Counter High
PCL	Program Counter Low
OPER	Operand
#	Immediate Addressing Mode

FIGURE 1. ASL-SHIFT LEFT ONE BIT OPERATION

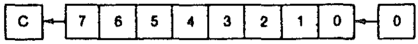


FIGURE 2. ROTATE ONE BIT LEFT (MEMORY OR ACCUMULATOR)

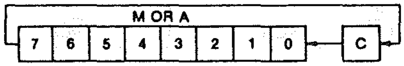
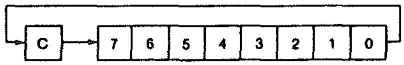


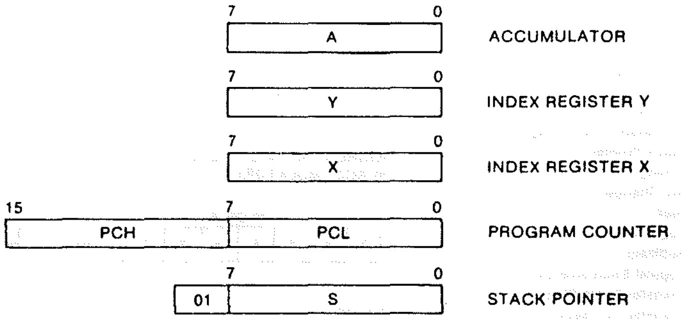
FIGURE 3.

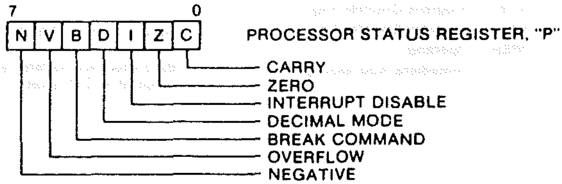


NOTE 1: BIT — TEST BITS

Bit 6 and 7 are transferred to the status register. If the result of A ∧ M is zero then Z=1, otherwise Z=0.

PROGRAMMING MODEL





INSTRUCTION CODES

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
ADC Add memory to accumulator with carry	$A + M + C \rightarrow A, C$	Immediate	ADC #Oper	69	2	√√√——√
		Zero Page	ADC Oper	65	2	
		Zero Page,X	ADC Oper,X	75	2	
		Absolute	ADC Oper	6D	3	
		Absolute,X	ADC Oper,X	7D	3	
		Absolute,Y	ADC Oper,Y	79	3	
		(Indirect,X)	ADC (Oper,X)	61	2	
		(Indirect),Y	ADC (Oper),Y	71	2	
AND "AND" memory with accumulator	$A \wedge M \rightarrow A$	Immediate	AND #Oper	29	2	√√————
		Zero Page	AND Oper	25	2	
		Zero Page,X	AND Oper,X	35	2	
		Absolute	AND Oper	2D	3	
		Absolute,X	AND Oper,X	3D	3	
		Absolute,Y	AND Oper,Y	39	3	
		(Indirect,X)	AND (Oper,X)	21	2	
		(Indirect),Y	AND (Oper),Y	31	2	
ASL Shift left one bit (Memory or Accumulator)	(See Figure 1)	Accumulator	ASL A	0A	1	√√√———
		Zero Page	ASL Oper	06	2	
		Zero Page,X	ASL Oper,X	16	2	
		Absolute	ASL Oper	0E	3	
		Absolute,X	ASL Oper,X	1E	3	
BCC Branch on carry clear	Branch on C=0	Relative	BCC Oper	90	2	——————
BCS Branch on carry set	Branch on C=1	Relative	BCS Oper	B0	2	——————
BEQ Branch on result zero	Branch on Z=1	Relative	BEQ Oper	F0	2	——————
BIT Test bits in memory with accumulator	$A \wedge M$, $M_7 \rightarrow N$, $M_6 \rightarrow V$	Zero Page	BIT* Oper	24	2	M_7√———M_6
		Absolute	BIT* Oper	2C	3	
BMI Branch on result minus	Branch on N=1	Relative	BMI Oper	30	2	——————
BNE Branch on result not zero	Branch on Z=0	Relative	BNE Oper	D0	2	——————
BPL Branch on result plus	Branch on N=0	Relative	BPL oper	10	2	——————
BRK Force Break	Forced Interrupt PC+2 ↓ P ↓	Implied	BRK*	00	1	———1——
BVC Branch on overflow clear	Branch on V=0	Relative	BVC Oper	50	2	——————

Note 1: [illegible] 6 and 7 are transferred to the status register. If the result of $A \wedge M$ is [illegible] then [illegible] 1 otherwise $Z = 0$

Note 2: A BRK command cannot be masked by setting I

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
BVS Branch on overflow set	Branch on V=1	Relative	BVS Oper	70	2	------
CLC Clear carry flag	0 → C	Implied	CLC	18	1	---0--
CLD Clear decimal mode	0 → D	Implied	CLD	D8	1	-0----
CLI	0 → I	Implied	CLI	58	1	---0--
CLV Clear overflow flag	0 → V	Implied	CLV	B8	1	0-----
CMP Compare memory and accumulator	A — M	Immediate	CMP #Oper	C9	2	√√√---
		Zero Page	CMP Oper	C5	2	
		Zero Page, X	CMP Oper,X	D5	2	
		Absolute	CMP Oper	CD	3	
		Absolute,X	CMP Oper,X	DD	3	
		Absolute,Y	CMP Oper,Y	D9	3	
		(Indirect,X)	CMP (Oper,X)	C1	2	
		(Indirect),Y	CMP (Oper),Y	D1	2	
CPX Compare memory and index X	X — M	Immediate	CPX #Oper	E0	2	√√√---
		Zero Page	CPX Oper	E4	2	
		Absolute	CPX Oper	EC	3	
CPY Compare memory and index Y	Y — M	Immediate	CPY #Oper	C0	2	√√√---
		Zero Page	CPY Oper	C4	2	
		Absolute	CPY Oper	CC	3	
DEC Decrement memory by one	M — 1 → M	Zero Page	DEC Oper	C6	2	√√----
		Zero Page,X	DEC Oper,X	D6	2	
		Absolute	DEC Oper	CE	3	
		Absolute,X	DEC Oper,X	DE	3	
DEX Decrement index X by one	X — 1 → X	Implied	DEX	CA	1	√√----
DEY Decrement index Y by one	Y — 1 → Y	Implied	DEY	88	1	√√----

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
EOR						
"Exclusive-Or" memory with accumulator	A V M → A	Immediate	EOR #Oper	49	2	√√----
		Zero Page	EOR Oper	45	2	
		Zero Page,X	EOR Oper,X	55	2	
		Absolute	EOR Oper	4D	3	
		Absolute,X	EOR Oper,X	5D	3	
		Absolute,Y	EOR Oper,Y	59	3	
		(Indirect,X)	EOR (Oper,X)	41	2	
		(Indirect),Y	EOR (Oper),Y	51	2	
INC						
Increment memory by one	M + 1 → M	Zero Page	INC Oper	E6	2	√√----
		Zero Page,X	INC Oper,X	F6	2	
		Absolute	INC Oper	EE	3	
		Absolute,X	INC Oper,X	FE	3	
INX						
Increment index X by one	X + 1 → X	Implied	INX	E8	1	√√----
INY						
Increment index Y by one	Y + 1 → Y	Implied	INY	C8	1	√√----
JMP						
Jump to new location	(PC+1) → PCL	Absolute	JMP Oper	4C	3	------
	(PC+2) → PCH	Indirect	JMP (Oper)	6C	3	
JSR						
Jump to new location saving return address	PC+2 ↓,	Absolute	JSR Oper	20	3	------
	(PC+1) → PCL					
	(PC+2) → PCH					
LDA						
Load accumulator with memory	M → A	Immediate	LDA #Oper	A9	2	√√----
		Zero Page	LDA Oper	A5	2	
		Zero Page,X	LDA Oper,X	B5	2	
		Absolute	LDA Oper	AD	3	
		Absolute,X	LDA Oper,X	BD	3	
		Absolute,Y	LDA Oper,Y	B9	3	
		(Indirect,X)	LDA (Oper,X)	A1	2	
		(Indirect),Y	LDA (Oper),Y	B1	2	
LDX						
Load index X with memory	M → X	Immediate	LDX #Oper	A2	2	√√----
		Zero Page	LDX Oper	A6	2	
		Zero Page,Y	LDX Oper,Y	B6	2	
		Absolute	LDX Oper	AE	3	
		Absolute,Y	LDX Oper,Y	BE	3	
LDY						
Load index Y with memory	M → Y	Immediate	LDY #Oper	A0	2	√√----
		Zero Page	LDY Oper	A4	2	
		Zero Page,X	LDY Oper,X	B4	2	
		Absolute	LDY Oper	AC	3	
		Absolute,X	LDY Oper,X	BC	3	

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
LSR Shift right one bit (memory or accumulator)	(See Figure 1)	Accumulator Zero Page Zero Page,X Absolute Absolute,X	LSR A LSR Oper LSR Oper,X LSR Oper LSR Oper,X	4A 46 56 4E 5E	1 2 2 3 3	0√√–––
NOP No operation	No Operation	Implied	NOP	EA	1	––––––
ORA "OR" memory with accumulator	A V M → A	Immediate Zero Page Zero Page,X Absolute Absolute,X Absolute,Y (Indirect,X) (Indirect),Y	ORA #Oper ORA Oper ORA Oper,X ORA Oper ORA Oper,X ORA Oper,Y ORA (Oper,X) ORA (Oper),Y	09 05 15 0D 1D 19 01 11	2 2 2 3 3 3 2 2	√√––––
PHA Push accumulator on stack	A ↓	Implied	PHA	48	1	––––––
PHP Push processor status on stack	P ↓	Implied	PHP	08	1	––––––
PLA Pull accumulator from stack	A ↑	Implied	PLA	68	1	√√––––
PLP Pull processor status from stack	P ↑	Implied	PLP	28	1	From Stack
ROL Rotate one bit left (memory or accumulator)	(See Figure 2)	Accumulator Zero Page Zero Page,X Absolute Absolute,X	ROL A ROL Oper ROL Oper,X ROL Oper ROL Oper,X	2A 26 36 2E 3E	1 2 2 3 3	√√√–––
ROR Rotate one bit right (memory or accumulator)	(See Figure 3)	Accumulator Zero Page Zero Page,X Absolute Absolute,X	ROR A ROR Oper ROR Oper,X ROR Oper ROR Oper,X	6A 66 76 6E 7E	1 2 2 3 3	√√√–––

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg N Z C I D V
RTI Return from interrupt	P↑ PC↑	Implied	RTI	40	1	From Stack
RTS Return from subroutine	PC↑, PC+1 → PC	Implied	RTS	60	1	------
SBC Subtract memory from accumulator with borrow	$A - M - \overline{C} \rightarrow A$	Immediate	SBC #Oper	E9	2	√√√--√
		Zero Page	SBC Oper	E5	2	
		Zero Page,X	SBC Oper,X	F5	2	
		Absolute	SBC Oper	ED	3	
		Absolute,X	SBC Oper,X	FD	3	
		Absolute,Y	SBC Oper,Y	F9	3	
		(Indirect,X)	SBC (Oper,X)	E1	2	
		(Indirect),Y	SBC (Oper),Y	F1	2	
SEC Set carry flag	1 → C	Implied	SEC	38	1	--1---
SED Set decimal mode	1 → D	Implied	SED	F8	1	----1-
SEI Set interrupt disable status	1 → I	Implied	SEI	78	1	---1--
STA Store accumulator in memory	A → M	Zero Page	STA Oper	85	2	------
		Zero Page,X	STA Oper,X	95	2	
		Absolute	STA Oper	8D	3	
		Absolute,X	STA Oper,X	9D	3	
		Absolute,Y	STA Oper,Y	99	3	
		(Indirect,X)	STA (Oper,X)	81	2	
		(indirect),Y	STA (Oper),Y	91	2	
STX Store index X in memory	X → M	Zero Page	STX Oper	86	2	------
		Zero Page,Y	STX Oper,Y	96	2	
		Absolute	STX Oper	8E	3	
STY Store index Y in memory	Y → M	Zero Page	STY Oper	84	2	------
		Zero Page,X	STY Oper,X	94	2	
		Absolute	STY Oper	8C	3	
TAX Transfer accumulator to index X	A → X	Implied	TAX	AA	1	√√----
TAY Transfer accumulator to index Y	A → Y	Implied	TAY	A8	1	√√----
TSX Transfer stack pointer to index X	S → X	Implied	TSX	BA	1	√√----

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
TXA Transfer index X to accumulator	X → A	Implied	TXA	8A	1	√√----
TXS Transfer index X to stack pointer	X → S	Implied	TXS	9A	1	------
TYA Transfer index Y to accumulator	Y → A	Implied	TYA	98	1	√√----

APPENDIX E

ACIA 6850 SPECIFICATIONS

This Appendix contains information on the 6850's architecture, operation and interface capabilities.

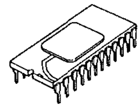
HD46850, HD468A50
ACIA (Asynchronous Communication Interface Adapter)

The HD46850 Asynchronous Communications Interface Adapter provides the data formatting and control to interface serial asynchronous data communications information to bus organized systems such as the HMCS6800 Microprocessing Unit.

The bus interface of the HD46850 includes select, enable, read/write, interrupt and bus interface logic to allow data transfer over an 8-bit bi-directional data bus. The parallel data of the bus system is serially transmitted and received by the asynchronous data interface, with proper formatting and error checking.

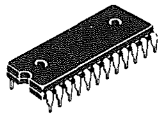
The functional configuration of the ACIA is programmed via the data bus during system initialization. A programmable Control Register provides variable word lengths, clock division ratios, transmit control, receive control, and interrupt control. For peripheral or modem operation three control lines are provided.

HD46850, HD468A50



(DC-24)

HD46850P, HD468A50P

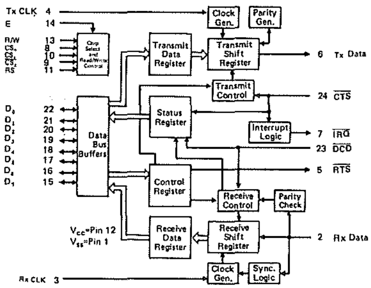


(DP-24)

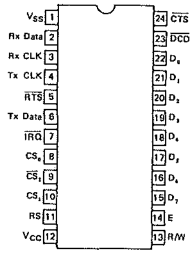
■ FEATURES

- Serial/Parallel Conversion of Data
- Eight and Nine-bit Transmission
- Insertion and Deleting of Start and Stop Bit
- Optional Even and Odd Parity
- Parity, Overrun and Framing Error Checking
- Peripheral/Modem Control Functions (Clear to Send $\overline{CTS}$, Request to Send $\overline{RTS}$, Data Carier Detect $\overline{DCD}$)
- Optional ÷ 1, ÷ 16, and ÷ 64 Clock Modes
- Up to 500kbps Transmission
- Programmable Control Register
- N-channel Silicon Gate Process
- Compatible with MC6850 and MC68A50

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}*	-0.3 ~ +7.0	V
Input Voltage	V_{in}*	-0.3 ~ +7.0	V
Operating Temperature	T_{opr}	-20 ~ +75	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol	min	typ	max	Unit
Supply Voltage	V_{CC}*	4.75	5	5.25	V
Input Voltage	V_{IL}*	-0.3	–	0.8	V
	V_{IH}*	2.0	–	V_{CC}	V
Operating Temperature	T_{opr}	-20	25	75	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

● DC CHARACTERISTICS (V_{CC}=5V±5%, V_{SS}=0V, Ta=-20~+75°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ*	max	Unit
Input "High" Voltage	All Inputs	V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage	All Inputs	V_{IL}		-0.3	–	0.8	V
Input Leakage Current	R/W, CS_0, CS_1, $\overline{CS_2}$, E	I_{in}	V_{in}=0~5.25V	–	–	2.5	μA
Three-State (Off State) Input Current	D_0~D_7	I_{TSI}	V_{in}=0.4~2.4V	–	–	10	μA
Output "High" Voltage	D_0~D_7	V_{OH}	I_{OH}=-205μA, Enable Pulse Width ≦ 25μs	2.4	–	–	V
	TxData, $\overline{RTS}$		I_{OH}=-100μA, Enable Pulse Width ≦ 25μs	2.4	–	–	
Output "Low" Voltage	All outputs	V_{OL}	I_{OL}=1.6mA, Enable Pulse Width ≦ 25μs	–	–	0.4	V
Output Leakage Current (Off State)	$\overline{IRQ}$	I_{LOH}	V_{OH}=2.4V	–	–	10	μA
Power Dissipation		P_D		–	300	525	mW
Input Capacitance	D_0~D_7	C_{in}	V_{in}=0V, T_a=25°C, f=1.0MHz	–	–	12.5	pF
	E, TxCLK, RxCLK, R/W, RS, RxData, CS_0, CS_1, $\overline{CS_2}$, $\overline{CTS}$, $\overline{DCD}$			–	–	7.5	
Output Capacitance	$\overline{RTS}$, TxData	C_{out}	V_{in}=0V, T_a=25°C, f=1.0MHz	–	–	10	pF
	$\overline{IRQ}$			–	–	5.0	

* T_a=25°C, V_{CC}=5V

- **AC CHARACTERISTICS**

1. TIMING OF DATA TRANSMISSION

Item		Symbol	Test Condition	min	typ	max	Unit
Minimum Clock Pulse Width	÷16, ÷64 Modes	PW_{CL}	Fig. 1	600	–	–	ns
	÷16, ÷64 Modes	PW_{CH}	Fig. 2	600	–	–	ns
Clock Frequency	÷1 Mode	f_c		–	–	500	kHz
	÷16, ÷64 Modes			–	–	800	
Clock-to-Data Delay for Transmitter		t_{TDD}	Fig. 3	–	–	1.0	μs
Receive Data Setup Time	÷ 1 Mode	t_{RDSU}	Fig. 4	500	–	–	ns
Receive Data Hold Time	÷ 1 Mode	t_{RDH}	Fig. 5	500	–	–	ns
$\overline{IRQ}$ Release Time		t_{IR}	Fig. 6	–	–	1.2	μs
$\overline{RTS}$ Delay Time		t_{RTS}	Fig. 6	–	–	1.0	μs
Rise Time and Fall Time	Except E	t_r, t_f		–	–	1.0*	μs

* 1.0 μs or 10% of the pulse width, whichever is smaller.

2. BUS TIMING CHARACTERISTICS

1) READ

Item	Symbol	Test Condition	HD46850			HD468A50			Unit
			min	typ	max	min	typ	max	
Enable Cycle Time	t_{cycE}	Fig. 7	1.0	–	–	0.666	–	–	μs
Enable "High" Pulse Width	PW_{EH}	Fig. 7	0.45	–	25	0.28	–	25	μs
Enable "Low" Pulse Width	PW_{EL}	Fig. 7	0.43	–	–	0.28	–	–	μs
Setup Time, Address and R/W valid to Enable positive transition	t_{AS}	Fig. 7	140	–	–	140	–	–	ns
Data Delay Time	t_{DDR}	Fig. 7	–	–	320	–	–	220	ns
Data Hold Time	t_H	Fig. 7	10	–	–	10	–	–	ns
Address Hold Time	t_{AH}	Fig. 7	10	–	–	10	–	–	ns
Rise and Fall Time for Enable Input	t_{Er}, t_{Ef}	Fig. 7	–	–	25	–	–	25	ns

2) WRITE

Item	Symbol	Test Condition	HD46850			HD468A50			Unit
			min	typ	max	min	typ	max	
Enable Cycle Time	t_{cycE}	Fig. 8	1.0	–	–	0.666	–	–	μs
Enable "High" Pulse Width	PW_{EH}	Fig. 8	0.45	–	25	0.28	–	25	μs
Enable "Low" Pulse Width	PW_{EL}	Fig. 8	0.43	–	–	0.28	–	–	μs
Setup Time, Address and R/W valid to Enable positive transition	t_{AS}	Fig. 8	140	–	–	140	–	–	ns
Data Setup Time	t_{DSW}	Fig. 8	195	–	–	80	–	–	ns
Data Hold Time	t_H	Fig. 8	10	–	–	10	–	–	ns
Address Hold Time	t_{AH}	Fig. 8	10	–	–	10	–	–	ns
Rise and Fall Time for Enable Input	t_{Er}, t_{Ef}	Fig. 8	–	–	25	–	–	25	ns

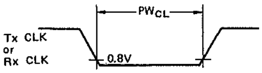


Figure 1 Clock Pulse Width, "Low" State

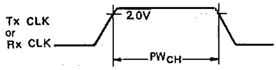


Figure 2 Clock Pulse Width, "High" State

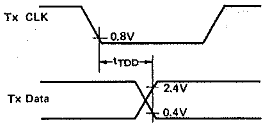


Figure 3 Transmit Data Output Delay

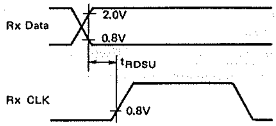


Figure 4 Receive Data Setup Time (÷1 Mode)

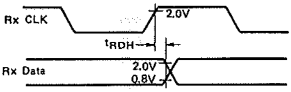


Figure 5 Receive Data Hold Time (÷1 Mode)

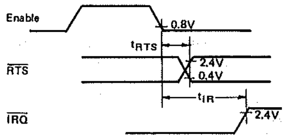


Figure 6 $\overline{RTS}$ Delay and $\overline{IRQ}$ Release Time

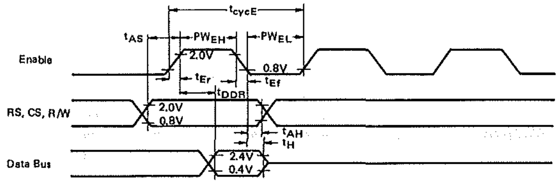


Figure 7 Bus Read Timing Characteristics
(Read information from ACIA)

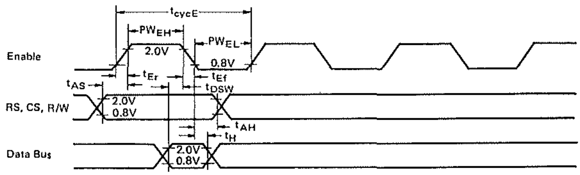
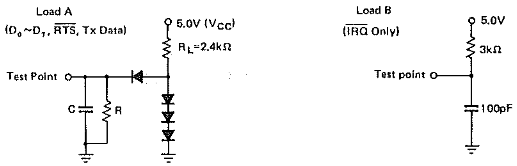


Figure 8 Bus Write Timing Characteristics
(Write information into ACIA)



C = 130pF for D_0~D_7
= 30pF for $\overline{RTS}$ and Tx Data
All diodes are 1S2074 Ⓗ or Equiv.

R = 11kΩ for D_0~D_7
= 24kΩ for $\overline{RTS}$ and Tx Data

Figure 9 Bus Timing Test Loads

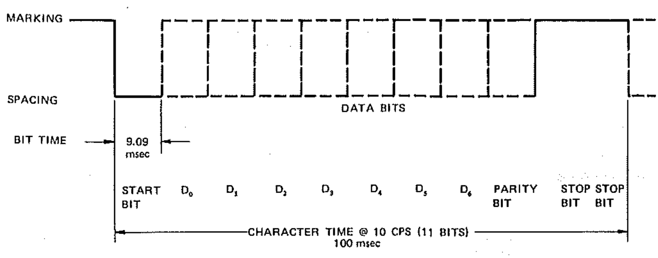
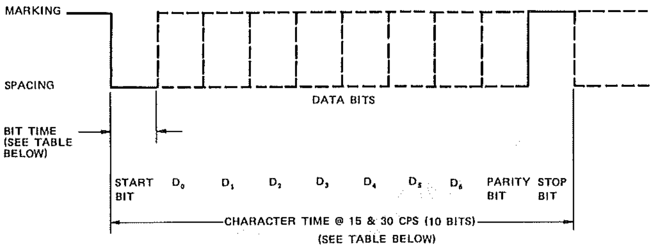


Figure 10 110 Baud Serial ASCII Data Timing



BAUD RATE	150	300
CHARACTERS/SEC	15	30
BIT TIME (msec)	6.67	3.33
CHARACTER TIME (msec)	66.7	33.3

$$\text{BIT TIME} = \frac{\text{SEC}}{\text{BAUD RATE}}$$

Figure 11 150 & 300 Baud Serial ASCII Data Timing

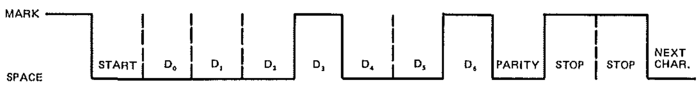


Figure 12 Send a 7 Bit ASCII Char. "H" Even Parity
– 2 Stop Bits H = 48_{16} = 1001000_2

■ DATA OF ACIA

HD46850 is an interface adapter which controls transmission and reception of Asynchronous serial data. Some examples of serial data are shown in Figs. 10 ~ 12.

■ INTERNAL STRUCTURE OF ACIA

HD46850(ACIA) provides the following; 8-bit Bi-directional Data Buses (D_0~D_7), Receive Data Input (Rx Data), Transmit Data Output (Tx Data), three Chip Selects (CS_0, CS_1, $\overline{CS_2}$), Register Select Input (RS), Two Control Input (Read/Write (R/W), Enable(E)), Interrupt Request Output($\overline{IRQ}$), Clear-to-Send ($\overline{CTS}$) to control the modem, Request-to-Send ($\overline{RTS}$), Data Carrier Detect($\overline{DCD}$) and Clock Inputs(Tx CLK, Rx CLK) used for synchronization of received and transmitted data. This ACIA also provides four registers; Status Register, Control Register, Receive Register and Transmit Register.

24-pin dual-in-line type package is used for the ACIA. Internal Structure of ACIA is illustrated in Fig. 13.

■ ACIA OPERATION

- **Master Reset**

The master reset (CR0, CR1) should be set during system initialization to insure the reset condition and prepare for programming the ACIA functional configuration when the communications channel is required. Control bits CR5 and CR6 should also be programmed to define the state of $\overline{RTS}$ whenever master reset is utilized. After master resetting the ACIA, the programmable Control Register can be set for a number of options such as variable clock divider ratios, variable word length, one or two stop bits, parity (even, odd, or none), etc.

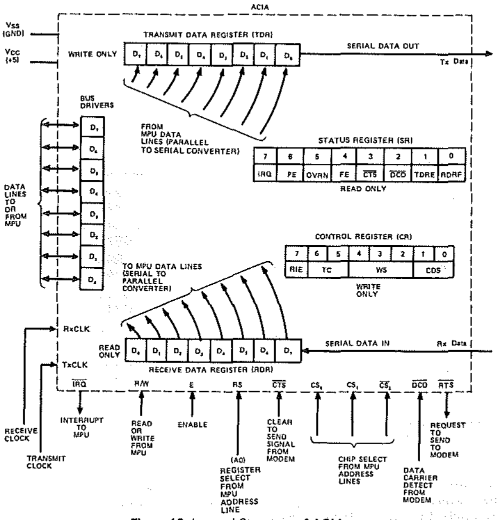


Figure 13 Internal Structure of ACIA

- **Transmit**

A typical transmitting sequence consists of reading the ACIA Status Register either as a result of an interrupt or in the ACIA's turn in a polling sequence. A character may be written into the Transmit Data Register if the status read operation has indicated that the Transmit Data Register is empty. This character is transferred to a Shift Register where it is serialized and transmitted from the Transmit Data output preceded by a start bit and followed by one or two stop bits. Internal parity (odd or even) can be optionally added to the character and will occur between the last data bit and the first stop bit. After the first character is written in the Data Register, the Status Register can be read again to check for a Transmit Data Register Empty condition and current peripheral status. If the register is empty, another character can be loaded for transmission even though the first character is in the process of being transmitted (because of double buffering). The second character will be automatically transferred into the Shift Register when the first character transmission is completed. This sequence continues until all the characters have been transmitted.

- **Receive**

Data is received from a peripheral by means of the Receive Data input. A divide-by-one clock ratio is provided for an externally synchronized clock (to its data) while the divide-by-16 and 64 ratios are provided for internal synchronization. Bit synchronization in the divide-by-16 and 64 modes is initiated by the detection of the leading mark-space transition of the start bit. False start bit detection capability insures that a full half bit of a start bit has been received before the internal clock is synchronized to the bit time. As a character is being received, parity (odd or even) will be checked and the error indication will be available in the Status Register along with framing error, overrun error, and Receive Data Register full. In a typical receiving sequence, the Status Register is read to determine if a character has been received from a peripheral. If the Receiver Data Register is full, the character is placed on the 8-bit ACIA bus when a Read Data command is received from the MPU. When parity has been selected for an 8-bit word (7 bits plus parity), the receiver strip the parity bit (D_7="0") so that data alone is transferred to the MPU. This feature reduces MPU programming. The Status Register can continue to be read again to determine when another character is available in the Receive Data Register. The receiver is also double buffered so that a character can be read from the data register as another character is being received in the Shift register. The above sequence continues until all characters have been received.

■ ACIA INTERNAL REGISTERS

The ACIA provides four registers; Transmit Data Register (TDR), Receive Data Register(RDR), Control Register(CR) and Status Register(SR). The content of each of the registers is summarized in Table 1.

Table 1 Definition of ACIA Register Contents

Buffer Address	RS=1 · R/W=0 ****	RS=1 · R/W=1	RS=0 · R/W=0	RS=0 · R/W=1
Data Bus	Transmit Data Register	Receiver Data Register	Control Register	Status Register
	(Write Only)	(Read Only)	(Write Only)	(Read Only)
0	Data Bit 0*	Data Bit 0	Counter Divide Select (CR0)	Rx Data Reg. Full (RDRF)
1	Data Bit 1	Data Bit 1	Counter Divide Select (CR1)	Tx Data Reg. Empty (TDRE)
2	Data Bit 2	Data Bit 2	Word Select 1 (CR2)	Data Carrier Detect ($\overline{DCD}$)
3	Data Bit 3	Data Bit 3	Word Select 2 (CR3)	Clear to Send ($\overline{CTS}$)
4	Data Bit 4	Data Bit 4	Word Select 3 (CR4)	Framing Error (FE)
5	Data Bit 5	Data Bit 5	Tx Control 1 (CR5)	Overrun (OVRN)
6	Data Bit 6	Data Bit 6	Tx Control 2 (CR6)	Parity Error (PE)
7	Data Bit 7***	Data Bit 7**	Rx Interrupt Enable (CR7)	Interrupt Request (IRQ)

* Leading bit = LSB = Bit 0
** Data bit will be zero in 7-bit plus parity modes.
*** Data bit is "don't care" in 7-bit plus parity modes.
**** 1 ... "High" level, 0 ... "Low" level

- **Transmit Data Register (TDR)**

Data is written in the Transmit Data Register during the negative transition of the enable (E) when the ACIA has been addressed and RS · $\overline{R/W}$ is selected. Writing data into the register causes the Transmit Data Register Empty bit in the Status Register to go "0". Data can then be transmitted. If the transmitter is idling and no character is being transmitted, then the transfer will take place within one bit time of the trailing edge of the Write command. If a character is being transmitted, the new data character will commence as soon as the previous character is complete. The transfer of data causes the Transmit Data Register Empty (TDRE) bit to indicate empty.

- **Receive Data Register (RDR)**

Data is automatically transferred to the empty Receive Data Register (RDR) from the receiver deserializer (a shift register) upon receiving a complete character. This event causes the Receive Data Register Full bit (RDRF) on the status buffer to go "1" (full). Data may then be read through the bus by addressing the ACIA and R/W "High" when the ACIA is enabled. The non-destructive read cycle causes the RDRF bit to be cleared to empty although the data is retained in the RDR. The status is maintained by RDRF as to whether or not the data is current. When the Receive Data Register is full, the automatic transfer of data from the Receiver Shift Register to the Data Register is inhibited and the RDR contents remain valid with its current status stored in the Status Register.

- **Control Register**

The ACIA Control Register consists of eight bits of write-only buffer that are selected when RS and R/W are "Low". This register controls the functon of the receiver, transmitter, interrupt enables, and the Request-to-Send ($\overline{RTS}$) peripheral/modem control output.

Counter Divide Select Bits (CR0 and CR1)

The Counter Divide Select Bits (CR0 and CR1) determine the divide ratios utilized in both the transmitter and receiver section of the ACIA. Additionally, these bits are used to provide a master reset for the ACIA which clears the Status Register (except for external conditions on $\overline{CTS}$ and $\overline{DCD}$) and initializes both the receiver and transmitter. Master reset does not affect other Control Register bits. Note that after power-on or a power fail/restart, these bits must be set "1" to reset the ACIA. After resetting, the clock divide ratio may be selected. These counter select bits provide for the following clock divide ratios:

Table 2 Function of Counter Devide Select Bit

CR1	CR0	Function
0	0	÷1
0	1	÷16
1	0	÷64
1	1	Master Reset

Word Select Bits (CR2, CR3, and CR4)

The Word Select bits are used to select word length, parity, and the number of stop bits. The encoding format is as follows:

Table 3 Function of Word Select Bit

CR4	CR3	CR2	Function
0	0	0	7 Bits + Even Parity + 2 Stop Bits
0	0	1	7 Bits + Odd Parity + 2 Stop Bits
0	1	0	7 Bits + Even Parity + 1 Stop Bit
0	1	1	7 Bits + Odd Parity + 1 Stop Bit
1	0	0	8 Bits + 2 Stop Bits
1	0	1	8 Bits + 1 Stop Bit
1	1	0	8 Bits + Even Parity + 1 Stop Bit
1	1	1	8 Bits + Odd Parity + 1 Stop Bit

Word length, Parity Select, and Stop Bit changes are not buffered and therefore become effective immediately.

Transmitter Control Bits (CR5 and CR6)

Two Transmitter Control bits provide for the control of the interrupt from the Transmit Data Register Empty condition, the Request-to-Send ($\overline{RTS}$) output, and the transmission of a Break level (space). The following encoding format is used:

Table 4 Function of Transmitter Control-Bit

CR6	CR5	Function
0	0	$\overline{RTS}$ = "Low", Transmitting Interrupt Disabled.
0	1	$\overline{RTS}$ = "Low", Transmitting Interrupt Enabled.
1	0	$\overline{RTS}$ = "High", Transmitting Interrupt Disabled.
1	1	$\overline{RTS}$ = "Low", Transmits a Break level on the Transmit Data Output. Transmitting Interrupt Disabled.

Receive Interrupt Enable Bit (CR7)

The following interrupts will be enabled by a "1" in bit position 7 of the Control Register (CR7): Receive Data Register Full, Overrun, or a "Low" to "High" transistion on the Data Carrier Detect ($\overline{DCD}$) signal line.

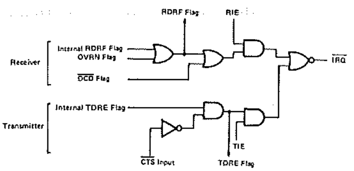


Fig. 14 $\overline{IRQ}$ Internal Circuit

- **Status Register**

Information on the status of the ACIA is available to the MPU by reading the ACIA Status Register. This read-only register is selected when RS is "Low" and R/W is "High". Information stored in this register indicates the status of the Transmit Data Register, the Receive Data Register and error logic, and the peripheral/modem status inputs of the ACIA.

Receive Data Register Full (RDRF), Bit 0

RDRF indicates that received data has been transferred to the Receive Data Register. RDRF is cleared after an MPU read of the Receive Data Register or by a master reset. The cleared or empty state indicates that the contents of the Receive Data Register are not current. Data Carrier Detect ($\overline{DCD}$) being "High" also causes RDRF to indicate empty.

Transmit Data Register Empty (TDRE), Bit 1

The Transmit Data Register Empty bit being set "1" indicates that the Transmit Data Register contents have been transferred and that new data may be entered. The "0" state indicates that the register is full and that transmission of a new character has not begun since the last write data command.

Data Carrier Detect ($\overline{DCD}$), Bit 2

The $\overline{DCD}$ bit will be "1" when the $\overline{DCD}$ input from a modem has gone "High" to indicate that a carrier is not present. This bit going "1" causes an Interrupt Request to be generated when the Receive Interrupt Enable is set. It remains "1" after the $\overline{DCD}$ input is returned "Low" until cleared by first reading the Status Register and then the Data Register or until a master reset occurs. If the $\overline{DCD}$ input remains "High" after read status and read data or master reset has occurred, the interrupt is cleared, the $\overline{DCD}$ status bit remains "1" and will follow the $\overline{DCD}$ input.

Clear-to-Send ($\overline{CTS}$), Bit 3

The $\overline{CTS}$ bit indicates the state of the $\overline{CTS}$ input from a modem. A "Low" $\overline{CTS}$ indicates that there is a $\overline{CTS}$ from the modem. In the "High state, the Transmit Data Register Empty bit is inhibited and the $\overline{CTS}$ status bit will be "1". Master reset does not affect the Clear-to-Send Status bit.

Framing Error (FE), Bit 4

FE indicates that the received character is improperly framed by a start and a stop bit and is detected by the absence of the 1st stop bit. This error indicates a synchronization error, faulty transmission, or a break condition. The FE flag is set or reset during the receive data transfer time. Therefore, this error indicator is present throughout the time that the associated character is available.

Receiver Overrun (OVRN), Bit 5

Overrun is an error flag that indicates that one or more characters in the data stream were lost. That is, a character or a number of characters were received but not read from the Receive Data Register (RDR) prior to subsequent characters being received. The overrun condition begins at the midpoint of the last bit of the second character received in succession without a read of the RDR having occurred. The overrun does not occur in the Status Register until the valid character prior to Overrun has been read. The RDRF bit remains set until the Overrun is reset. Character synchronization is maintained during the Overrun condition. The Overrun indication is reset after the reading of data from the Receive Data Register or by a Master Reset.

Parity Error (PE), Bit 6

The PE flag indicates that the number of "1"s (highs) in the character does not agree with the preselected odd or even parity. Odd parity is defined to be when the total number of ones is odd. The parity error indication will be present as long as the data character is in the RDR. If no parity is selected, then both the transmitter parity generator output and the receiver parity check results are inhibited.

Interrupt Request (IRQ), Bit 7

The IRQ bit indicates the state of the $\overline{IRQ}$ output. Any interrupt condition with its applicable enable will be indicated in this status bit. Anytime the $\overline{IRQ}$ output is "Low" the IRQ bit will be "1" to indicate the interrupt or service request status. IRQ is cleared by a read operation to the Receive Data Register or a write operation to the Transmit Data Register.

■ SIGNAL FUNCTIONS

● Interface Signal for MPU

Bi-Directional Data Bus (D_0~D_7)

The bi-directional data bus (D_0~D_7) allow for data transfer between the ACIA and the MPU. The data bus output drivers are three-state devices that remain in the high impedance (off) state except when the MPU performs an ACIA read operation.

Enable (E)

The Enable signal, E, is a high impedance TTL compatible input that enables the bus input/output data buffers and clocks data to and from the ACIA. This signal will normally be a derivative of the HMCS6800 ϕ_2 Clock.

Read/Write (R/W)

The R/W line is a high impedance input that is TTL compatible and is used to control the direction of data flow through the ACIA's input/output data bus interface. When R/W is "High" (MPU Read cycle), ACIA output drivers are turned on and a selected register is read. When it is "Low", the ACIA output drivers are turned off and the MPU writes into a selected register. Therefore, the R/W signal is used to select read-only or write-only registers within the ACIA.

Chip Select (CS_0, CS_1, $\overline{CS_2}$)

These three high impedance TTL compatible input lines are used to address the ACIA. The ACIA is selected when CS_0 and CS_1 are "High" and $\overline{CS_2}$ is "Low". Transfers of data to and from the ACIA are then performed under the control of the Enable signal, Read/Write, and Register Select.

Register Select (RS)

The RS line is a high impedance input that is TTL compatible. A "High" level is used to select the Transmit/Receive Data Registers and a "Low" level the Control/Status Registers. The R/W signal line is used in conjunction with Register Select to select the read-only or write-only register in each register pair.

Interrupt Request ($\overline{IRQ}$)

$\overline{IRQ}$ is a TTL compatible, open-drain (no internal pullup), active "Low" output that is used to interrupt the MPU. The $\overline{IRQ}$ output remains "Low" as long as the cause of the interrupt is present and the appropriate interrupt enable within the ACIA is set.

Clock Inputs

Separate high impedance TTL compatible inputs are provided for clocking of transmitted and received data. Clock frequencies of 1, 16 or 64 times the data rate may be selected.

Transmit Clock (Tx CLK)

The Tx CLK input is used for the clocking of transmitted data. The transmitter initiates data on the negative transition of the clock.

Receive Clock (Rx CLK)

The Rx CLK input is used for synchronization of received data. (In the ÷ 1 mode, the clock and data must be synchronized externally.) The receiver samples the data on the positive transition of the clock.

● Serial Input/Output Lines

Receive Data (Rx Data)

The Rx Data line is a high impedance TTL compatible input through which data is received in a serial format. Synchronization with a clock for detection of data is accomplished internally when clock rates of 16 or 64 times the bit rate are used. Data rates are in the range of 0 to 500 kbps when external synchronization is utilized.

Transmit Data (Tx Data)

The Tx Data output line transfers serial data to a modem or other peripheral. Data rates in the range of 0 to 500 kbps when external synchronization is utilized.

Modem Control

The ACIA includes several functions that permit limited control of a peripheral or modem. The functions included are $\overline{CTS}$, $\overline{RTS}$ and $\overline{DCD}$.

Clear-to-Send ($\overline{CTS}$)

This high impedance TTL compatible input provides automatic control of the transmitting end of a communications link via the modem $\overline{CTS}$ active "Low" output by inhibiting the Transmit Data Register Empty (TDRE) status bit.

Request-to-Send ($\overline{RTS}$)

The $\overline{RTS}$ output enables the MPU to control a peripheral or modem via the data bus. The $\overline{RTS}$ output corresponds to the state of the Control Register bits CR5 and CR6. When CR6=0 or both CR5 and CR6=1, the $\overline{RTS}$ output is "Low" (the active state). This output can also be used for Data Terminal Ready ($\overline{DTR}$).

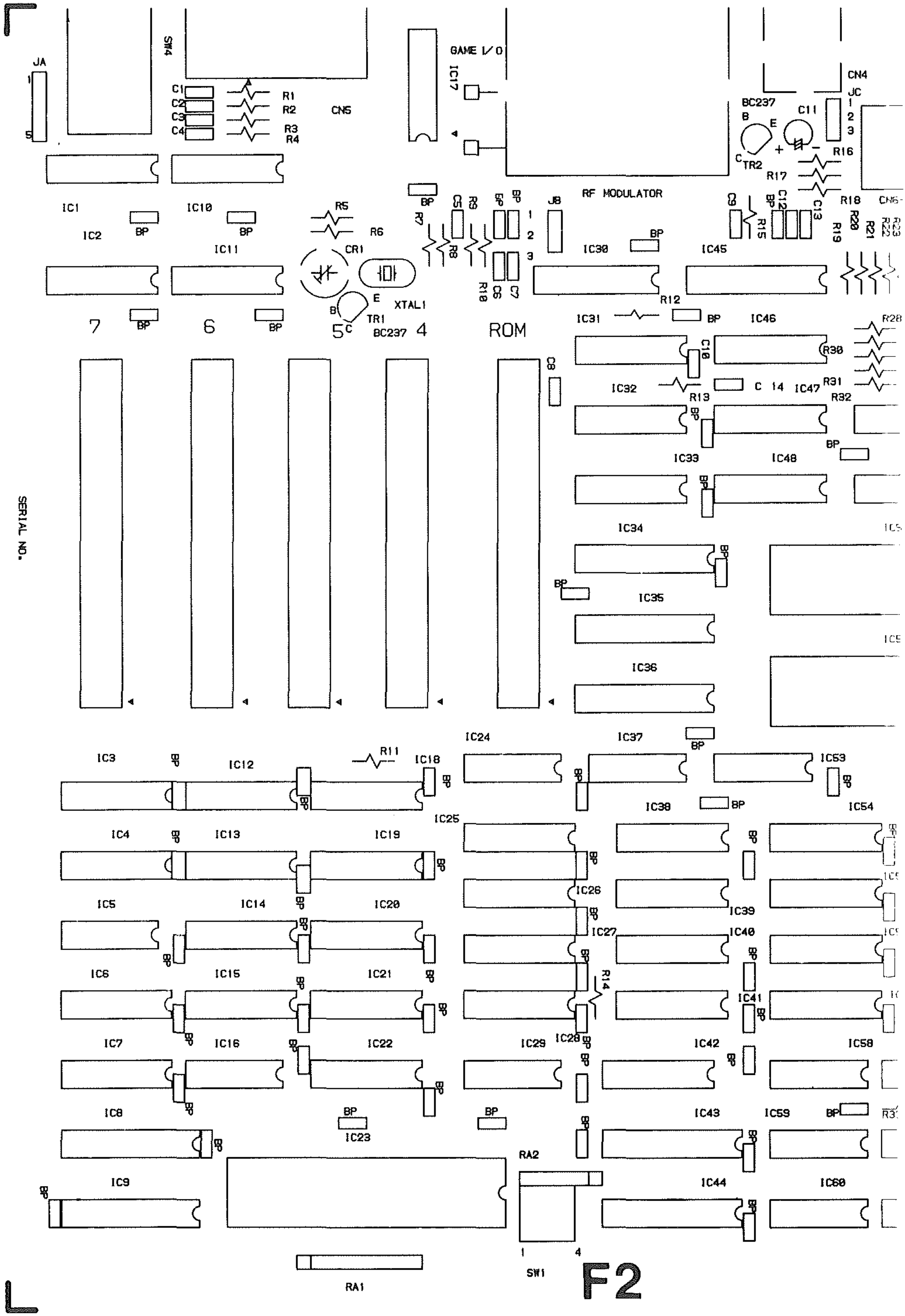
Data Carrier Detect ($\overline{DCD}$)

This high impedance TTL compatible input provides automatic control, such as in the receiving end of a communications link by means of a modem $\overline{DCD}$ output. The $\overline{DCD}$ input inhibits and initializes the receiver section of the ACIA when "High". A "Low" to "High" transition of the $\overline{DCD}$ initiates an interrupt to the MPU to indicate the occurrence of a loss of carrier when the Receive Interrupt Enable bit is set.

APPENDIX F

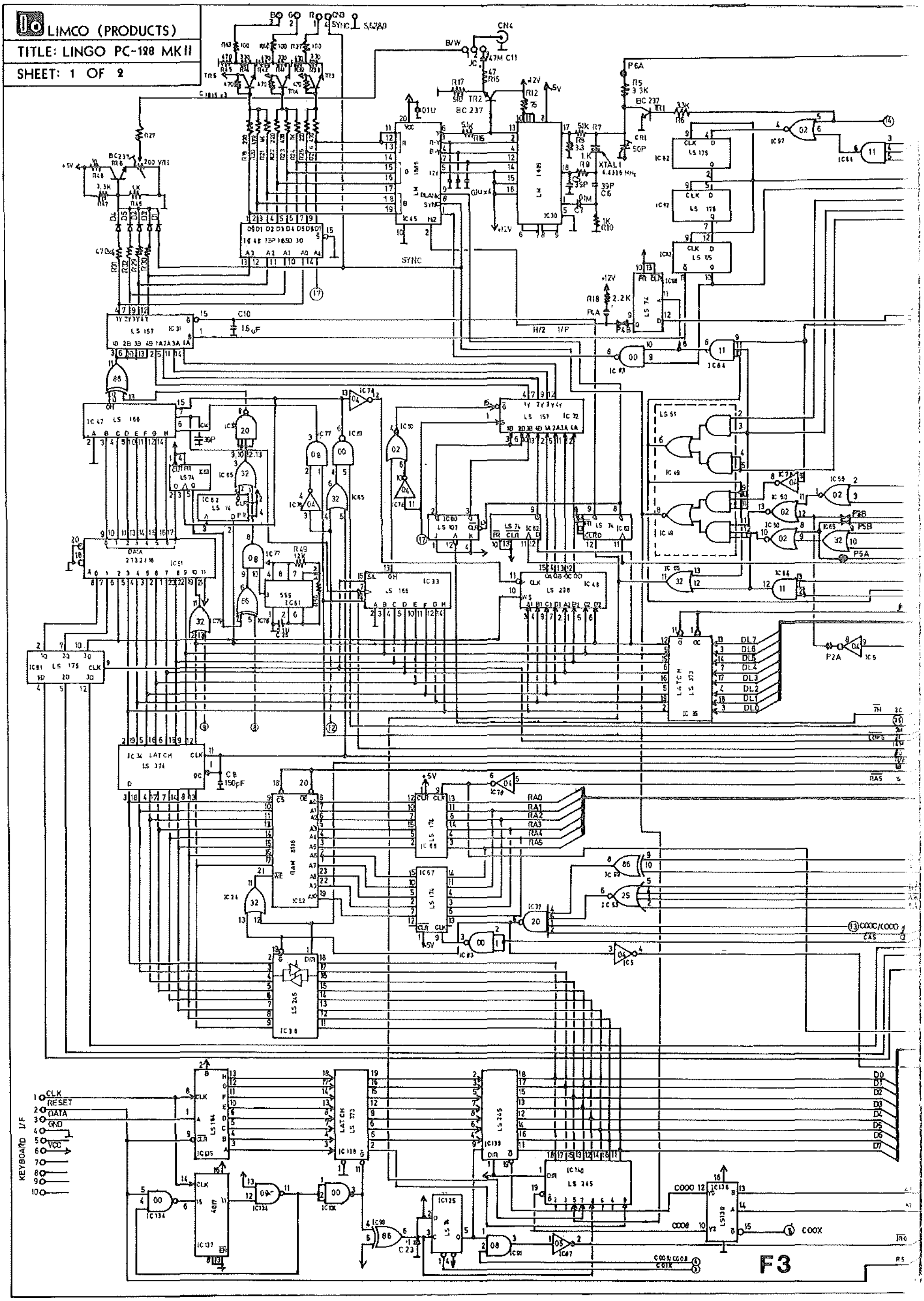
SCHEMATICS

This Appendix contains the System Board layout and schematic diagrams.



JA
SW4
GAME I/O
IC17
CN4
JC
C1 C2 C3 C4
R1 R2 R3 R4
CN5
BC237
C11
TR2
R16
R17
RF MODULATOR
R18
IC1
IC10
IC2
IC11
R5
R6
CR1
R7
R8
R9
C5
C6
C7
R10
JB
C9
R15
C12
C13
R19 R20 R21 R22 R23
IC30
IC45
XTAL1
TR1
BC237
7
6
5
4
ROM
IC31
R12
IC46
R28
C8
C10
R30
IC32
R13
C 14
IC47
R31
R32
IC33
IC48
SERIAL NO.
IC34
IC35
IC36
IC24
IC37
IC3
IC12
R11
IC18
IC53
IC38
IC54
IC25
IC4
IC13
IC19
IC26
IC39
IC5
IC14
IC20
IC27
IC40
IC6
IC15
IC21
R14
IC41
IC7
IC16
IC22
IC29
IC28
IC42
IC58
IC8
IC23
IC43
IC59
IC9
RA2
IC44
IC60
RA1
SW1
BP
F2

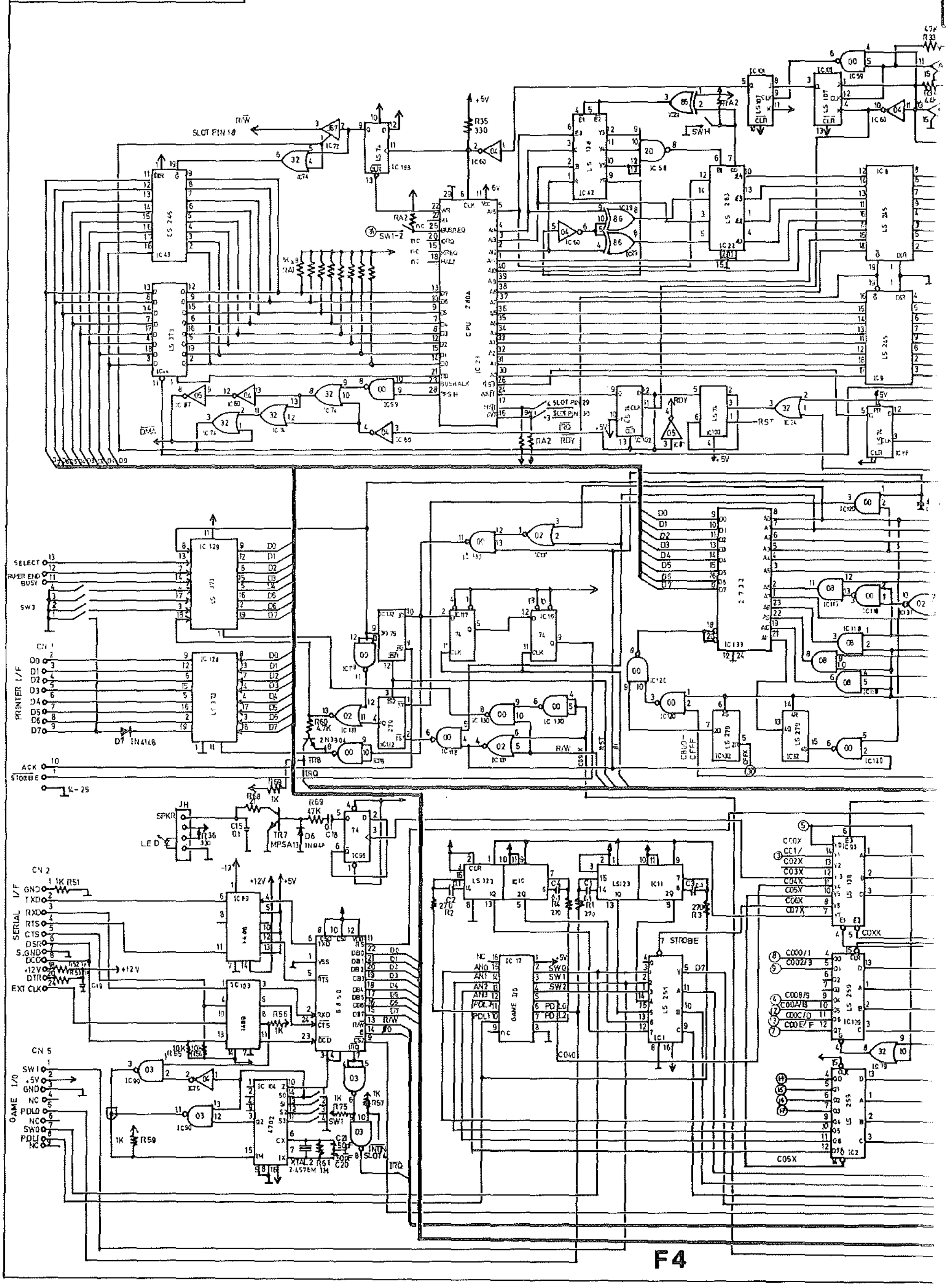
LIMCO (PRODUCTS)
TITLE: LINGO PC-128 MK II
SHEET: 1 OF 2



F3

TITLE: LINGO PC-128 MK II

SHEET: 2 OF 2



F4

APPENDIX G

OPTIONAL PERIPHERAL AND DEVICES

SAM CARD

The SAM (Software Automatic Mouth) card is a versatile speech synthesizer that adds quality speech to your LINGO PC−128 mkII. This comes complete with a programmed diskette and a manual.

MUSIC SYNTHESIZER CARD

Create music in stereo with 9 separate channels of voices. Great music skill is not necessary.

256K RAM CARD

Reduce your disk access time by installing a 256K pseudo RAM disk to your LINGO PC−128 mkII.

RGB COLOR MONITOR

Add high resolution colours to your LINGO PC−128 mkII computer system.

PX−120 DOT MATRIX PRINTER

A versatile printer for your text and graphic printing. Features includes near−letter−quality text mode, 1K on−board− memory buffer, friction and adjustable sprocket feedings.

OTHER PRODUCTS

Hard disk, joystick, diskette containers, diskettes, power pad educational kit, etc.

See your nearest LINGO dealer for more information.

LIMCO (PRODUCTS) MFG. PTE. LTD.