

## UPCOMING TOPICS

- Putting assembly language into perspective
- Overview of built-in system firmware
  - Drivers
  - Interrupts
- Assembler Lab #2 – using MS-DOS function calls
- Programming examples – hardware-specific programs
- Developing custom ROMs

PPA4A010



Notes:

PPA4A010

## PROGRAMMING APPROACHES

- High-level languages
  - Easiest to develop
  - Portable code
  
- Assembly language
  - Small program size
  - Fastest execution
  
- High-level with low-level routines
  - Easier development
  - Speed where required
  - Access to more features

PPA4C010



### Notes:

PPA4C010

Choose a programming approach which is appropriate for the task at hand--

Will the program be used on other machines?

Is execution speed a critical factor?

Is hardware cost a concern (ROM size, edisc size)?

Do you need access to features not supported by the language?

## MORE ON (moron?) PROGRAMMING

	High-level language	Assembly language
HP escape sequences	Most transportable across HP	Convenience features
ANSI escape sequences	IBM PC compatibility	
MS-DOS function calls	MS-DOS transportability	
BIOS interrupts	Additional functionality	Fast, limited transportability
Direct to hardware		Fastest, no transportability

PPA4C020



### Notes:

PPA4C020

Most features may be accessed from more than one level--choose a level which gives the necessary combination of speed and transportability.

## PORTABLE PLUS DRIVERS

### Serial Devices

AUX	redirectable
COM1	internal RS-232
COM2=82164A	external RS-232
COM3	internal modem

### Printers

PRN	redirectable
LPT1	first HP-IL printer
LPT2=LST	second HP-IL printer

### Miscellaneous

PLT	redirectable plotter
CLOCK	
NUL	
CON	

PPA4E010



### Notes:

PPA4E010

Drivers are similar to the 110 except:

- COM3 is new. The modem and RS-232 are independent and can be used simultaneously.
- CON is expanded.

## ESCAPE SEQUENCE SUMMARY

	<i>Control Codes</i>	<i>HP Two-Character</i>	<i>HP Parameterized</i>	<i>ANSI</i>
<i>Total</i>	11	22	52	20
<i>New</i>	Font select (~N, ~O)	Reset terminal  Display functions	Relative cursor positioning  Arabic mode Auto line feed Bell enable Keyboard modes Graphical	Page change
<i>Improved</i>			Underline, halfbright Local/xmit softkeys Cursor select Return serial number	Underline, halfbright Multiple insert or delete More display modes
<i>Changed</i>			25-line screen 62-line page Model 45711	

PPA4E020



### Notes:

PPA4E020

The Portable PLUS has about 30% more escape sequences than the 110, plus others which have been improved. This allows better terminal emulation.

Half of the parameterized escape sequences are for graphics and did not exist on the 110.

While the graphics escape sequences are quite complete, this machine cannot fully emulate an HP graphics terminal because graphics and alpha are mutually exclusive.

## KEYBOARD MODES

- HP mode                      Compatible with HP terminals
- Alternate mode              Compatible with IBM PC
- Modifier mode              One- or two-character keycodes. Modifier keys generate keycodes on both up and down transition
- Scancode mode              Raw keycodes only--no ASCII conversion, uppercasing, etc.
- Numeric keypad mode       Combines with any of the first three modes



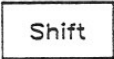

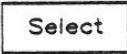
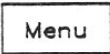
PPA4E030



### Notes:

PPA4E030

## EXAMPLES: KEYBOARD MODES

Keys Pressed	Codes Generated (Interrupt Taken)			
	HP	Alternate	Modifier	Scancode
	1B 70 esc p	00 3B nul ;	9B 70 hesc p	01
	3B ;	3B ;	3B ;	61D
 	3A :	3A :	82 3A 80 shift : unshift	61D
	1B 26 50 esc & P	none	8E	40D
	(Int 56K)	00 43 nul C	8C	17D

PPA4E040



### Notes:

PPA4E040

All modes except Scancode have multi-byte keys. In general, HP multi-byte keys begin with escape, IBM keys begin with nul, and Modifier keys begin with hescape.

Scancode cannot distinguish shifted, extended, or control characters. To decode the modifier keys, you must use the modifier keys hook.

## WHAT IS AN INTERRUPT?

- INT instruction is a far call
- IRET instruction is a far return
- Far call addresses
  - Reside in lowest 1K of memory
  - 4 bytes for each of 256 interrupts

PPA4G010



### Notes:

PPA4G010

INT and IRET work just like CALL and RET, except the call address is defined by the operating system, not the program.



## EXAMPLE: SERVICE INTERRUPT

### MS-DOS Function Call

- INT 21H
  - Push flags, CS, IP
  - Load IP from 0:84H
  - Load CS from 0:86H
- Code at new CS:IP checks the contents of AH and executes the function call
- IRET pops IP, CS, flags
- User program resumes with next instruction

PPA4G020



### Notes:

PPA4G020

The interrupt instruction saves three of the registers; other registers used by the function call are saved by that call.

## EXAMPLE: INTERRUPT HOOK

### Menu Key Hook

- If:
  - HP keyboard mode
  - **Menu** key gets pressed
  
- Then:
  1. Console driver generates INT 56H
  2. Flags, CS, IP pushed on stack
  3. CS, IP loaded from 0:158H
  4. Code at CS:IP toggles the menu, and
  5. Does IRET to console driver
  
- Menu key behavior may be redefined by far address at 0:158H

PPA4G030



### Notes:

PPA4G030

Service interrupts are generated by the user's program; hook interrupts are generated by the operating system.

## INTERRUPT SUMMARY

5H	}	IBM PC services and hooks
1FH	}	MS-DOS services and hooks
20H	}	Hooks for hardware interrupts
3FH	}	HP services and hooks
40H	}	Available for applications
4FH	}	Hardware interrupt
50H	}	
5FH	}	
60H	}	
FEH	}	
FFH	}	

PPA4G040



### Notes:

PPA4G040

The 256 interrupts are arranged in groups, where only the MS-DOS services and hooks are required for an MS-DOS machine.

We emulate most of the IBM BIOS interrupts for compatibility's sake, but in general, the HP services are more powerful and faster.

### IBM PC BIOS INTERRUPTS

- Two hooks: timer tick, keyboard break

- Ten service routines

#### Examples:

- 10H Video input/output
  - \* Differs from IBM version due to resolution
  - \* Alpha read/write in graphics mode
  - \* Other functions similar to fast video interrupt
- 14H Communications interrupt
  - \* Low-level access to AUX device
  - \* 82164A not supported

PPA4G050



#### Notes:

PPA4G050

Probably the most useful IBM interrupt is INT 10H because of its ability to do alpha in graphics mode. The other interrupts provide low-level access to essentially every hardware device, but they are seldom the best option available.

## MS-DOS INTERRUPTS

	<i>Interrupts</i>		<i>AH Functions</i>
	21H function request	→	01H read keyboard and echo
			02H display character
			03H input character from AUX
			4DH retrieve child's return code
	22H } 23H } exit addresses 24H }		
	25H absolute disc read		
	26H absolute disc write		
	27H terminate, stay resident		

PPA4G060



### Notes:

PPA4G060

Most MS-DOS capabilities are accessed through interrupt 21H, where AH must contain the number of the desired function. Low-numbered functions are older low-level routines, while higher-numbered functions are more sophisticated and easier to use.

## SYSTEM SERVICES INT 50H

### ■ Machine-specific utilities

- ROM select
- Font loading
- PPU access

### ■ Machine-optimized utilities

- Display string
- Display character

### ■ New utilities

- Simulate keyboard input

PPA4G070



### Notes:

PPA4G070

This and all following interrupts are unique to this computer.

System services should be used when possible to avoid direct access to the hardware (for compatibility with future products).

## EXPANSION HOOKS

- CON expansion 5EH
  - Called at display initialization, processing character, etc.
  - Usable to add escape sequences or filter characters
  
- AUX expansion 5DH
  - Called during serial or modem interrupt
  - Usable for handshaking or filtering

PPA4G080



### Notes:

PPA4G080

The expansion hooks allow you to modify the behavior of a standard driver without totally replacing it. Both CON and AUX can be expanded.

## FAST VIDEO INT 5FH

- 43 fast alpha functions
  - Operate on logical rectangle
  - Characters and/or attributes
  - Cursor—relative or rectangle—relative
  
- 15 fast graphics functions
  - Lines and dots
  - Area fill, read, write
  - Graphics cursor

PPA4G090



### Notes:

PPA4G090

Fast video is the most powerful set of functions for display control. Alpha functions operate in a "logical rectangle" which is some subset of alpha memory. The display window is independent of the LR, so the LR or some portion may not be displayed. The cursor is always in the LR.

The console driver actually uses fast video calls to manipulate the display.



## HP-IL PRIMITIVES INT 54H

- Relatively fast
- I/O may be interleaved
- Controller only

PPA4G100



### Notes:

PPA4G100

The HP-IL primitives are used for custom I/O to instruments or other devices not supported by the system software.

Users of this interrupt need HP-IL expertise. See:

The HP-IL System: An Introductory Guide  
Kane/Harper/Ushijima  
Osborne/McGraw-Hill

HP 82166-90016 HP-IL Integrated Circuit Manual  
HP 82166-90017 HP-IL Interface Specification

## ASSEMBLER LAB #2

### ■ Overview

Using MS-DOS function calls, write a program that reads characters from the keyboard, uppercases them, then displays them.

### ■ *Extra credit*

*Provide for uppercasing of ROMAN8 characters as well as ASCII characters.*

PPA4I010



Notes:

PPA4I010

## LAB #2 - BACKGROUND

- Function 8: Wait for a key to be pressed. If it is control C, end program; otherwise, return the character in AL.
  
- Function 2: Display the character in DL.
  
- *Function 38H: Return country-specific information.*

PPA41020



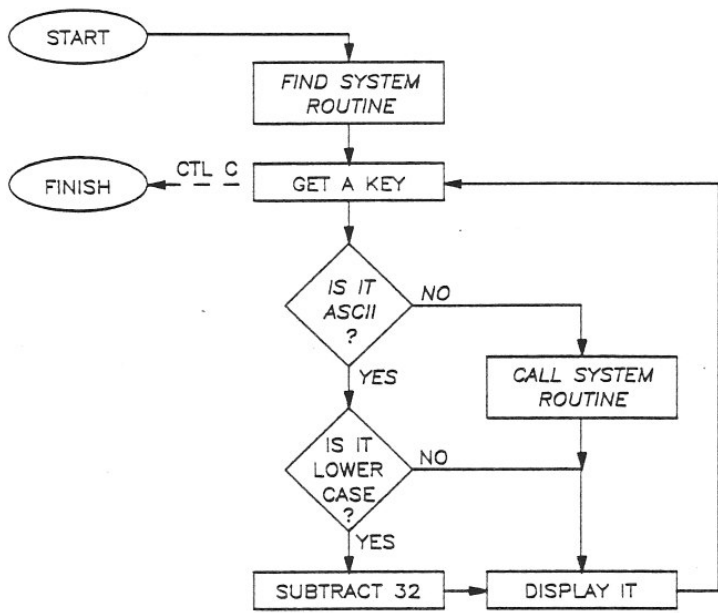
### Notes:

PPA41020

Consult your System Programmer's Guide (from the Programmer's Toolkit) for more information.

# Module 4: System Software

## LAB #2 FLOWCHART



PPA4I030



Notes:

PPA4I030

## HARDWARE-SPECIFIC PROGRAMMING EXAMPLES

- Custom key labels
- Using HP-IL primitives
- Forms input using fast video

PPA4K010



**Notes:**

PPA4K010

## CUSTOM KEY LABELS

- Display controller automatically "locks" key labels to bottom of screen.
- Display controller can lock 0, 1, 2, or 3 lines.
- Menu key interrupt can be redirected.

PPA4K020



### Notes:

PPA4K020

See the example program.

## USING HP-IL PRIMITIVES

- Typical code sequence
  - Configure loop
  - Find device
  - Talker and listener addresses
  - Set timeout
  - Send or receive, string or frame
  
- Other capabilities
  - Get accessory ID
  - Read interface status

PPA4K040



### Notes:

PPA4K040

Configure, find device, and addressing are ignored if the loop has been used in the last 30 seconds and the addresses given are valid.

See the example program.

## FORMS INPUT USING FAST VIDEO

- Position and clear logical rectangle
- Position window
- Write text and forms
- Position cursor
- Read keyboard
- Check for special characters
- Check for form boundaries
- Display character

PPA4K060



### Notes:

PPA4K060

See the example program.



## CUSTOM ROM SOFTWARE

- ROM addressing
- ROM contents
  - ROM disc
  - ROM-executable
- ROM development tools
  - ROM simulator
  - EPROMs in ROM drawer
- ROM simulator lab

PPA4M010

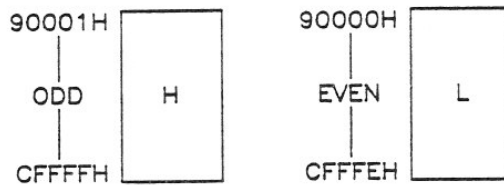


Notes:

PPA4M010

## ROM ADDRESSING

- Six banks up to 256K each
- Each bank is a pair of ROMs:



- All ROMs contain:
  - Boot sector
  - FAT
  - Directory (30 entries)
  - At least one ROM-disc file

PPA4M020



Notes:

PPA4M020

## ROM-DISC PROGRAMS

- Just like a disc except faster
- No special programming required
- Any configuration
  - 32K            Half-bank
  - 32K x 2      Full-bank
  - 128K          Half-bank
  - 128K x 2     Full-bank

PPA4M030



### Notes:

PA4M030

The memory management software takes care of reading alternate bytes from half-bank ROMs; a half-bank ROM may be located in any slot.

Full-bank ROMs must be installed side-by-side in H and L slots.

## ROM-EXECUTABLE PROGRAMS

- Advantages
  - Large programs do not use system RAM
  - Best program security
  - ROM in bank seven can redefine system
- ROM-executable code follows all ROM-disc files in ROM
- Must be a full-bank ROM pair
- Special programming required
  - No embedded variables
  - ROM-disc program to enable ROM, jump to it

PPA4M040



Notes:

PPA4M040

## CUSTOM ROM DEVELOPMENT

1. Create program(s).
2. Use IMAGE utility to create ROM image in ROM SIMULATOR.
3. Use burner utility (MDS) and EPROM BURNER to create EPROMs for field testing.
4. Buy hard ROMs.

PPA4M050



**Notes:**

PPA4M050

## ROM SIMULATOR

- Not a product!
- 512K NMOS RAM emulates ROM banks 0,1
- Separate power supply
- RAM/ROM switch
- Use IMAGE utility to load it

PPA4M060



### Notes:

PPA 4M060

The advantage of the ROM simulator is that it eliminates frequent burning of EPROMs during the development stage.

## EPROMs IN ROM DRAWER

- Drawer reconfigured with jumpers
- Accepts 32K CMOS EPROMs
- May contain both ROMs and EPROMs
  - 0/12
  - 4/8
  - 8/4
  - 12/0
- Eight EPROMs emulate full 256K bank
- MDS utility assumes EPROM burner conforming to Intel's MDS (microcomputer development system) standard
- Only EPROM burner tested: Data I/O 29A

PPA4M070



### Notes:

PPA4M070

## ROM SIMULATOR LAB

1. Assemble several favorite programs on a single disc (under 256K).
2. Load them into the ROM simulator  
IMAGE [<responsefile>]
3. Verify that they work.

PPA4M080



Notes:

PPA4M080