HEWLETT hp PACKARD

6940B Multiprogrammer System Throughput Analysis

for Multiprogrammer Systems using the 9825A Desktop Computer

Application Note 282-1

TABLE OF CONTENTS

MARK
777-8085

Chapter I
INTRODUCTION

1-1 The purpose of this analysis is to give the 6940B Multiprogrammer user an idea of how much time the various I/O transfers involving the 9825A desk-top computer and the 6940B require. With this analysis it is possible to tell if the 6940B can handle a given application at all, and if so, the kind of timing to be expected.

1-2 Two methods of interfacing the 9825A to the 6940B exist: 16 bit duplex, also known as GPIO (98032A option 040); and the HP-IB[1] (98034A). Program coding and timing are given for both methods of interfacing.

1-3 Although this analysis will serve to qualify or disqualify the 6940B for most applications, a few warnings must be observed. First of all, the times contained herein cannot be taken as absolute specifications. Rather they are guidelines around which variations will occur due to variations in temperature, equipment age, and calculator options.

1-4 The analysis has been done on a modular basis. That is, times are given for modules such as high speed output or low speed input. If a user desired to couple several modules together into one operation, the times will, in most cases, add linearly. The user should be careful though, since in some cases, due to interaction, the times will not add. See Appendix B for additional information on this topic.

1-5 In most circumstances the user will encounter times that are within 10% of the times called out herein. If the user's requirement falls within this 10% margin of the specified time, extreme care must be exercised.

1-6 In order to use this guide, the appropriate 6940B I/O Structure is first selected by reading Section II of this analysis. If the user is programming the high speed structures he must also add in the additional card time as called out in figure 5-1. The additional card time is the time a given Multiprogrammer plug-in card takes over the automatic handshake time. The automatic handshake time is explained in the 6940B Users Guide for the 9825A (59500-90005).

For example, if a user were programming a 69321B D/A Card with the High Speed output structure with GPIO, the total time would be .050 ms for the transfer plus .015 ms additional card time for a total of about .065 ms.

Throughput Summary Chart (average speed in milliseconds)

Operation	16 Bit Duplex GPIO	HP-IB	See Section
INPUT			
Library	32	54	III
Medium Speed	2.0	3.7	IV
High Speed (buffered)	.03	—	V
Analog Scan	56 channels/sec	50 channels/sec	IV
OUTPUT			
Library	30	46	III
Medium Speed	1.6	2.8	IV
High Speed (buffered)	.04	.15	V

[1]HP-IB is Hewlett-Packard's implementation of IEEE 488-1975 (ANSI standard MC1.1).

Chapter II
THE THREE 6940B, 9825A PROGRAM STRUCTURES

2-1 Depending on the type of I/O data transfers a programmer must perform, he will choose one of three I/O structures: Software Library, Medium Speed or High Speed.

2-2 The Software Library is intended for systems where speed is not an important criterion. If a system can tolerate transfers that require up to 100ms each (see table 3-1), the system programmer may make use of the 6940B-9825A Software Library (HP Model No. 14556A). By employing the Library, the programmer need not delve into the workings of the 6940B, but can control I/O operations at a high level. For example, the subprogram "Analog Input" takes care of the Multiprogrammer control, address, and return data words and also converts the input voltage into a decimal value.

2-3 For systems requiring somewhat faster throughput rates, in the neighborhood of one to five milliseconds each (see table 4-1), the medium speed structure can be used. This is the method used by programmers who are familiar with the 6940B, and whose programs will intermix I/O operations and computations.

2-4 When the I/O operations must be done at high speeds of up to 25,000 transfers per second, the high speed structure is used. The buffered block transfer operations of the 9825A are used to achieve these speeds (see table 5-2). This structure is only useful when the I/O operations are not to be intermixed with computations involving these I/O values. That is, the program can be broken into two distinct segments: (1) the computing of input or output values and, (2) the block transfer.

2-5 As an example of the high speed structure, take the case where a digital recorder is to be tested at its recording speed of 20,000 words per second. Here a fast read/write buffer in the 9825 can be filled with the test pattern which can then be sent at high speed to the recorder through the 6940B.

2-6 An example of high speed input is waveform digitization. Here a waveform is fed into the fast A/D card which digitizes the analog signal. The digital result is put in a buffer in the 9825A. When the buffer is full, the digitized waveform can be examined.

2-7 High speed transfers can be intermixed with operations not involving the I/O data by making use of the interrupt or DMA type buffers. In this manner a user can overlap his I/O transfers with background calculations, as long as he does not attempt to use any data involved in the I/O transfer.

Chapter III
SOFTWARE LIBRARY

3-1 The 6940B-9825A Software Library is designed for use in cases where the primary consideration is to get the system up and running quickly. By sacrificing operating speed, the programmer can put together a working system with relative ease. The Software Library enables the programmer to run the 6940B without getting involved at a machine language level. Refer to Table 3-1 to determine if the throughput speeds are sufficient for a given application.

3-2 The following is an example of using the Software Library (note that the autoloading feature of the Library has automatically loaded the required subprograms after the mainline program.

User Writes This Code (lines 0–12)

Auto Loader Appends These Subroutines Automatically (lines 13–62)

```
0: "Software Library Example":
1:
2:
3:
4: cll 'Voltage Serial Output'(1,A,2,B)
5: cll 'Parallel Output'(3,C)
6: cll 'Analog Input'(4,D)
7:
8:
9:
10:
11:
12: end
13: "Parameter Check":
14: "PC":if flg10#1;mdec;dev "MP",9;wti 0,9;sfg 10
15: fmt 9,c,f2.0,/,c,f2.0,2/
16: if int(p0/2)=p0/2;1+p0→p0;0→pp0
17: fxd 0;moct;1→p39
18: if pp39>=400 and pp39<=414;pp39-400→pp39;gsb +9
19: if pp39>=0 and pp39<=14;jmp 2
20: dsp "Slot Address p",p39,"out of range";stp ;jmp -2
21: if p(p39+1)<=7777;jmp 2
22: dsp "Data, p",p39+1,"out of range";stp ;jmp -1
23: if (p39+2→p39)<=p0-1;gto -5
24: if pp0<=15;jmp 2
25: dsp "Unit no.",pp0,"out of range";stp ;jmp -1
26: ret
27: wrt 16.9,"Parameter p",p39,"changed to ",pp39;ret
28: "Voltage Serial Output":
29: "VS":gsb "Parameter Check"
30: wtb "MP",170160+dtopp0
31: 1→p39
32: pp39→p(p39+40);if frc(pp39)#0;gto "QUAD"
33: if p(p39+1)>10.235 or p(p39+1)<-10.24;beep;fxd 0;gto +10
34: p(p39+1)/.005→p(p39+41);if p(p39+41)<0;p(p39+41)+4096→p(p39+41)
35: wtb "MP",10000+dtop(p39+40)+dtop(p39+41)
36: if (p39+2→p39)<=p0-1;gto -4
37: wtb "MP",170040;mdec
38: ret
39: "QUAD":if p(p39+1)>10.22 or p(p39+1)<-10.24;sfg 9;gto +4
40: p(p39+1)/.02→p(p39+41);if p(p39+41)<0;p(p39+41)+1024→p(p39+41)
41: (10*frc(p(p39+40))-1)*1024+p(p39+41)→p(p39+41)
42: prnd(pp39,0)→p(p39+40);jmp -7
43: dsp "Data, p",p39+1,"out of range";stp ;gto -10;if flg9;cfg 9;gto -4
44: "Parallel Output":
45: "PO":gsb "Parameter Check"
46: wtb "MP",170040+dtopp0
47: 1→p39
48: wtb "MP",10000+dtopp39+p(p39+1)
49: if (p39+2→p39)<=p0-1;gto -1
50: wtb "MP",170160,170040;mdec
51: ret
52: "Analog Input":
53: "AI":gsb "Parameter Check"
54: wtb "MP",170260+dtopp0
55: 1→p39
56: wti 4,10000+dtopp39;rdb("MP")→p(p39+1)
57: if p(p39+1)>7777;p(p39+1)-100000→p(p39+1)
58: otdp(p39+1)→p(p39+1);if p(p39+1)>2047;p(p39+1)-4096→p(p39+1)
59: .005p(p39+1)→p(p39+1)
60: if (p39+2→p39)<=p0-1;gto -4
61: wtb "MP",170040;mdec
62: ret
*5293
```

GPIO Software Library Example

User Writes This Code

```
0: "Software Library Example":
1:
2:
3:
4: cll 'Voltage Serial Output'(1,A,2,B)
5: cll 'Parallel Output'(3,C)
6: cll 'Analog Input'(4,D)
7:
8:
9:
10:
11:
12: end
```

Auto Loader Appends These Subroutines Automatically

```
13: "Parameter Check":
14: "PC":if flg10#1;dev "MP",723,"HP-IB",7;sfg 10
15: fmt 9,c,f2.0,/,c,f2.0,2/
16: if int(p0/2)=p0/2;1+p0→p0;0→pp0
17: 1→p39
18: if pp39>=400 and pp39<=414;pp39-400→pp39;gsb +9
19: if pp39>=0 and pp39<=14;jmp 2
20: dsp "Slot address p",p39," out of range";stp ;jmp -2
21: if p(p39+1)<=7777;jmp 2
22: dsp "Data,p",1+p39,"out of range";stp ;jmp -1
23: if (p39+2→p39)<=p0-1;gto -5
24: if pp0<=15;jmp 2
25: dsp "Unit no.",pp0,"out of range";stp ;jmp -1
26: fmt 1,c,z;fmt 2,c,f4.0,c,z;ret
27: wrt 16.9,"Parameter p",p39,"changed to ",pp39;ret
28: "Voltage Serial Output":
29: "VS":gsb "Parameter Check"
30: wrt "MP.2","O",160+dtopp0,"T"
31: 1→p39
32: pp39→p(p39+40);if frc(pp39)#0;gto "QUAD"
33: if p(p39+1)>10.235 or p(p39+1)<-10.24;beep;fxd 0;gto +10
34: p(p39+1)/.005→p(p39+41);if p(p39+41)<0;p(p39+41)+4096→p(p39+41)
35: wrt "MP.2",char(64+p(p39+40)),dtop(p39+41),"T"
36: if (p39+2→p39)<=p0-1;gto -4
37: wrt "MP.1","00040T"
38: ret
39: "QUAD":if p(p39+1)>10.22 or p(p39+1)<-10.24;sfg 9;gto +4
40: p(p39+1)/.02→p(p39+41);if p(p39+41)<0;p(p39+41)+1024→p(p39+41)
41: (10*frc(p(p39+40))-1)*1024+p(p39+41)→p(p39+41)
42: prnd(p(p39+40),0)→p(p39+40);jmp -7
43: dsp "Data, p",p39+1,"out of range";stp ;gto -10;if flg9;cfg 9;gto -4
44: "Parallel Output":
45: "PO":gsb "Parameter Check"
46: wrt "MP.2","O",40+dtopp0,"T"
47: 1→p39
48: wrt "MP.2",char(64+pp39),p(p39+1),"T"
49: if (p39+2→p39)<=p0-1;gto -1
50: wrt "MP.1","00160T00040T"
51: ret
52: "Analog Input":
53: "AI":gsb "Parameter Check"
54: wrt "MP.2","O",260+dtopp0,"T"
55: 1→p39
56: wrt "MP.2",char(64+pp39),"T",char(64+pp39),"X";red "MP",p(p39+1)
57: if p(p39+1)>7777;p(p39+1)-10000→p(p39+1)
58: otdp(p39+1)→p(p39+1);if p(p39+1)>2047;p(p39+1)-4096→p(p39+1)
59: .005p(p39+1)→p(p39+1)
60: if (p39+2→p39)<=p0-1;gto -4
61: wrt "MP.1","00040T"
62: ret
*11911
```

HP-IB Software Library Example

3-3 From table 3-1 it can be determined how long the subprograms called in the example will take to execute. In line 4, of the GPIO example, the "Voltage Serial Output" with 2 data parameters will require 20 ms + 2 x 31 ms for a total of 82 ms. Likewise, the line 5 "Parallel Output" call requires 20 ms + 18 ms or 38 ms total. The "Analog Input" call of line 6 requires a total of 51 ms.

Table 3-1. Software Library Summary[1] (N=the number of data parameters)

Program Name		GPIO	HP-IB
Arm	(AM)	20 + 17N	25 + 18N
Analog Input	(AI)	20 + 31N	25 + 35N
Current Serial Output	(CS)	23 + 22N	25 + 25N
Current Parallel Output	(CP)	21 + 23N	28 + 23N
Clear Unit	(CU)	92	120
Double Ended Scan	(DE)	61 + 54N	70 + 72N
Disarm	(DM)	23 + 16N	24 + 18N
Interrupt Output	(IO)[2]	28 + 18N	3 + 20N
Parallel Output	(PO)	20 + 18N	25 + 20N
Parallel Input	(PI)	7 + 74N	29 + 82N
Parameter Check	(PC)	11 + 7N	15 + 11N
Read Counter	(RC)	18 + 20N	23 + 25N
Relay Output	(RO)	19 + 37N	21 + 46N
Recycle 69600B	(RE)	20 + 18N	25 + 20N
Reference Word	(RW)	23 + 21N	17 + 27N
Simple Output	(oo)	34	40
Serial Output	(SO)	18 + 12N	25 + 20N
Simple Input	(ii)	34	45
Serial Input	(SI)	16 + 14N	24 + 28N
Single Ended Scan	(SE)	52 + 36N	57 + 50N
Set Counter	(SC)	19 + 22N	23 + 24N
Serial Poll	(SP)	19 + 19N	24 + 22N
Voltage Serial Output	(VS)	20 + 31N	25 + 32N
Voltage Parallel Output	(VP)	20 + 31N	24 + 33N

1. The times are expressed as a constant plus the number of data parameters times another constant, eg.:
   The time to input the three analog values in c l l 'AI' (1,A,2,B,3,C) is 20 + 31 x 3 = 113ms.
2. The "wait 1000" statement has been removed in this case. If the "wait 1000" is left in, the time will be 880 ms longer.

Chapter IV
MEDIUM SPEED

4-1 In cases where the user wishes to increase the throughput rate by programming the 6940B on a lower level than the Software Library, the Medium Speed structure is used. The Medium Speed structure is used whenever higher speeds are required than can be obtained using the Software Library. If additional speed is required above the Medium Speed rates, the High Speed structure may be the solution.

4-2 To use the Medium Speed structure, the user must be familiar with programming the 6940B at a machine language level. This familiarity may be gained by reading the 6940B Multiprogrammer Users Guide for the HP 9825A.

4-3 In this section certain 9825A instructions are contained in parentheses. This is code executed at a point prior to the transfer code execution. Since it need not be executed every time a transfer is to take place, the user will probably not want to include it in the input or output time calculation.

4-4 OUTPUT

4-5 General

4-6 To send an octal variable to any card the programmer must send a control word and a data word:

GPIO: [1](moct) (94 μs)
wtb9, 170160, 20000 + A (1.9 ms)

HP-IB: [1](fmt 1, "0160TB", f4.0, "T", z) (350 μs)
wrt 723.1,A (3.0 ms)

Where A contains the octal data and the data is sent to slot 402. The fact that a variable was sent instead of a constant cost 0.3ms on GPIO and 0.7ms on HP-IB.

If, instead of the simple variable A, we send the following, then the following allowances must be made:

dtoA — allow an additional 0.5 ms
D [A] — allow an additional 0.2 ms

See Appendix 1 for format considerations on the HP-IB

4-7 Data Word Only. In cases where the correct control word has previously been given to the 6940B, only the data word need be sent:

GPIO: [1](moct, control word)
wtb9, 20000 + A (1.4 ms)

HP-IB: [1](fmt 2, "B", f4.0, "T"; Control Word)
wrt 723.2, A (2.7 ms)

4-8 Analog

4-9 In order to program an analog output card, the conversion from decimal voltage to two's complement notation must be made:

dto (X*200 + 4096 (X < 0))→A (2.1 ms)

where X = desired voltage in volts
A = volts in 2's complement LSB's and
1LSB = 5mV

This requires 2.1 ms. Otherwise, the speeds are the same as for digital output.

4-10 INPUT

4-11 General

4-12 The input procedure requires three steps. A control word followed by an address word must be sent to the 6940B and the return data word must be read:

GPIO: (moct; wti 0,11) (780 μs)
wtb9, 170260; wti4, 20000;
rdb(9)→A (3.0 ms)

HP-IB: (fmt 2, "0260T", c,z) (290 μs)
wrt 723.2, "BTX"; red 723,A (4.7 ms)

In these examples, the input data from the card in slot 402 is put into the variable A.

If a second input is desired from the same card:

GPIO: rdb (9)→A (1.0 ms)

HP-IB: (fmt1,c,z)
wrt 723.1,"X"; red 723,A (4.3 ms)
[2]live date — red 723,A (2.7 ms)

[1]Program code in parentheses is code executed sometime before the transfer is to take place.
[2]To read live data through the 59500A a "Z" is used as the terminator of the address word, e.g., 'wrt 723.1, "AZ".'

In cases where the correct control word has previously been sent to the 6940B and an input from a different card is desired, only the address word need be sent before reading the data word.

GPIO: wti4,20000; rdb (9)—→A (1.9 ms)

HP-IB: (fmt1,c,z)
wrt 723.1, "CTX"; red 723,A (4.3 ms)

4-13 Analog

4-14 As in the output case, the same procedures and times are involved for taking single readings from the analog input card (69422A). However, it may be desirable to convert the two's complement data into decimal:

((otdX→X) - 4096 (X>2047)) *.005→A (2.3 ms)

X = reading from Analog card in 2's complement LSB's. 1LSB = 5mV.

A = reading in decimal volts

4-15 Multichannel Analog Scanning: Up to twelve channels of analog input are possible with one 69433A Relay with Readback Card and one 69422A A/D Converter Card. If a double ended scan is required, two 69433A Cards will supply 12 inputs. For more inputs (up to 168 single ended or 84 double ended) more relay cards can be added. For still more inputs, extender mainframes may be added to the system.

4-16 Twelve Channel Single Ended (wired as in Figure 4-1)

GPIO: Scan rate = 56 channels/sec

```
0: "Scan SE....GPIO...69433A in slot 10; A/D in slot 9":
1: dim D[1000];moct;wti 0,11
2: for N=1 to 12;wtb 9,170160,120000,120000+shf(1,1-N)
3: wtb 9,170260;wti 4,110000;rdb(9)→D[N];next N
4: end
*25958
```

HP-IB: Scan rate = 50 channels/sec

```
0: "Scan SE...HP-IB...69433A in slot 10; A/D in slot 9":
1: fmt 2,c,f4.0,c,z;dim D[2000]
2: for N=1 to 12;wrt 723.2,"0160TJTJ",shf(1,1-N),"T0260TITX"
3: red 723,D[N];next N
4: end
*18395
```

Line 1 dimensions an array to hold input data and, for GPIO, sets the 9825A I/O address register to select code 9 (octal 11).

Line 2 is the first part of the for-next loop. The write command sends a control word with TME, DTE, and SYE on. All the relays are opened to assure no two are closed at the same time. One relay at a time is then closed.

For GPIO line 3 is the second part of the for-next loop. The control word here turns on ISL and TME in preparation for input. The address word for slot 409 is now sent. For HP-IB, this is done at the end of line 2.

The data is read with gate into an array by the read binary instruction, for GPIO, and for HP-IB by writing "T" and reading with "red". The for-next loop is repeated 12 times; once for each of the twelve relays.

In these examples the 69433A is in slot 410 and the 69422A is in slot 409.

4-17 Before each scan all relays are opened to assure that no two relays are ever shorted together. If a relay were programmed to close in the same step as one were programmed to open, there is the possibility that the closure may occur first. Note that these routines only read the two's complement data from the analog input cards, they do not convert the data to decimal values. The programmer may choose to read in all the data and then convert the array of data at a later time. Alternatively the programmer may choose to make use of the dead time while the relays are opening to do the conversions.

4-18 Twelve Channel Single Ended Scan with Conversion

GPIO: 56 channels/sec

```
0: "Scan and Convert SE...GPIO   69433A in slot 10, A/D in slot 9":
1: dim D[1000];moct;wti 0,11
2: for N=1 to 12;wtb 9,170160,120000+shf(1,1-N),170260
3: wti 4,110000;rdb(9)→X;wtb 9,170140,120000
4: ((otdX→X)-4096(X>2047))*200→D[N];next N
5: end
*23661
```

The program is similar to Scan SE GPIO, but now the relays are all opened up at the end of line 3. While the relays open, program execution continues to line 4 where the two's complement to decimal conversion takes place. The program waits at line 2 for the relay card to time out before closing the next relay.

HP-IB: 50 channels/sec

```
0: "Scan and Convert SE...HP-IB   69433A in slot 10, A/D in slot 9":
1: fmt 2,c,f4.0,c,z;fmt 1,c,z;dim D[2000]
2: for N=1 to 12;wrt 723.2,"O160TJ",shf(1,1-N),"TO260TITX"
3: red 723,X;wrt 723.1,"O140TJT";((otdX→X)-4096(X>2047))*200→D[N];next N
4: end
*28545
```

This program is similar to Scan SE HP-IB, but now the relays are all opened in line 3. While the relays open, the 9825A does the two's complement to decimal conversion. The program waits at line 2 for the relay card to time out before closing the next relay.

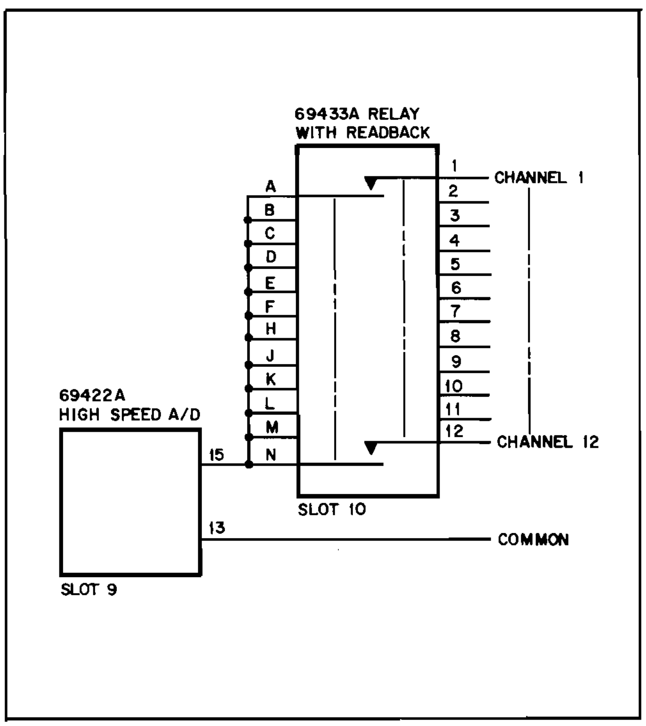

Figure 4-1. Hardward Setup for Single Ended Scan

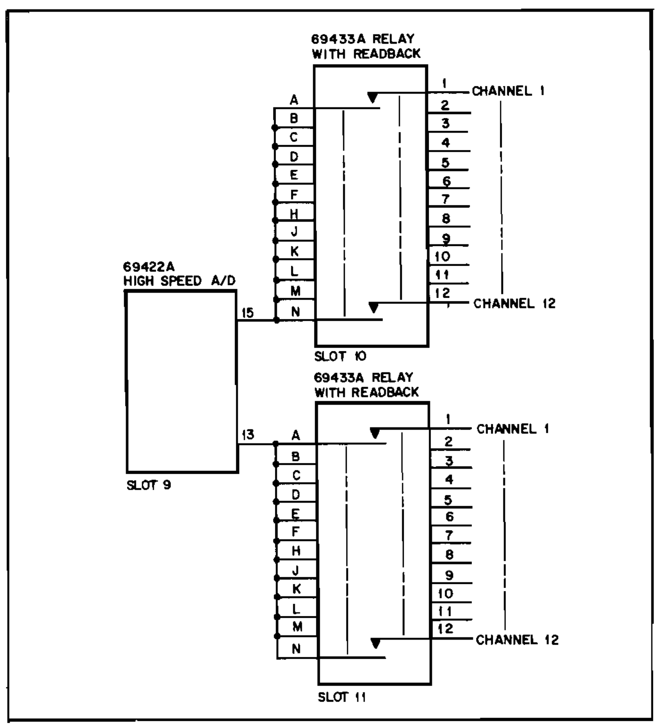

Figure 4-2. Hardware Setup for Double Ended Scan

4-19 In these examples the 69433A is in slot 10 and the 69422A is in slot 9. The conversions in this case are done while relays are opening in preparation for the next scan at a considerable savings in overall system time.

4-20 Double Ended Scan. In cases where the common terminals of the scanned voltages cannot be wired together, a double ended scan will be required. Figure 4-2 shows how to wire a 12 channel double ended scanner.

scan rate without conversion

GPIO = 55 channels/sec

```
0: "Scan DE....GPIO...69433A in slots 10&11; A/D in slot 9":
1: dim D[1000];moct;wti 0,11
2: for N=1 to 12;wtb 9,170140,120000,130000,170160
3: shf(1,1-N)→P
4: wtb 9,170140,120000+P,130000+P,170260
5: wti 4,110000;rdb(9)→D[N];next N
6: end
*28188
```

HP-IB = 35 channels/sec

```
0: "Scan DE...HP-IB...69433A in slots 10&11; A/D in slot 9":
1: fmt 1,c,z;fmt 3,c,f4.0,c,f4.0,c,z;dim D[2000]
2: for N=1 to 12;wrt 723.1,"0140TJTKT0160T"
3: shf(1,1-N)→P
4: wrt 723.3,"0140TJ",P,"TK",P,"T0260TITX"
5: red 723,D[N];next N
6: end
*2497
```

Line 1 dimensions an array to hold the input data and, for GPIO sets the 9825A I/O address register to select code 9 (octal 11).

Line 2 begins the for-next loop of the scan. First a control word is sent with DTE and SYE on. This is so the two data words that follow to open all relays can be sent at automatic handshake speeds. After the data words have been sent, TME is turned on to prevent further 6940B programming until the relay cards have timed out.

Line 3 sets a bit in the variable P corresponding to the relays to be closed on each of the two relay cards for the next scan.

Line 4 again sends a control word with TME off to put the 6940B in automatic handshake mode. The proper relays are then closed. A control word with TME on is sent to guarantee that the relays will have finished closing before the 6940B begins its next step. For HP-IB the address word for slot 409 (the A/D) is now sent.
The data read from the A/D card with gate is stored in an array.

The for-next loop is repeated 12 times; once for each pair of relays.

scan rate with conversion

GPIO = 55 channels/sec

```
0: "Scan and Convert DE...GPIO   69433A in slots 10&11; A/D in slot 9":
1: dim D[1000];moct;wti 0,11
2: for N=1 to 12;shf(1,1-N)→P
3: wtb 9,170140,120000+P,130000+P,170260;wti 4,110000
4: rdb(9)→X;wtb 9,170140,120000,130000,170160
5: ((otdX→X)-4096(X>2047))*200→D[N];next N
6: end
*28081
```

HP-IB = 35 channels/sec

```
0: "Scan and Convert DE...HP-IB   69433A in slots 10&11; A/D in slot 9":
1: fmt 1,c,z;fmt 3,c,f4.0,c,f4.0,c,z;dim D[2000]
2: for N=1 to 12;shf(1,1-N)→P
3: wrt 723.3,"0140TJ",P,"TK",P,"T0260TITX"
4: red 723,X;wrt 723.1,"0140TJTKT0160T"
5: ((otdX→X)-4096(X>2047))*200→D[N];next N
6: end
*21112
```

This program is similar to Scan DE, but now the relays are opened in line 4. While the relays open, the two's complement reading is converted into decimal. The program then waits at line 3 until the relay cards time out before programming the next relay closures.

4-21 Example of Medium Speed Timing

```
0: "Medium Speed Example":
1: ent "value to be programmed", V
2: moct
3: dto (V * 200 + 4096 (V< 0)) → V
4: wtb 9, 170160, 40000 + V
5: end
```

In this example line 4 will require 1.9 ms. The conversion on line 3 will take 2.1 ms as shown on page 4-1

Table 4-1. Medium Speed Summary

Operation	16 Bit Duplex	HP-IB	Text Reference Paragraph
OUTPUT:	Time: 1.4 — 1.9 ms Rate: 714 — 526 (per sec)	2.7 — 3.0 ms 370 — 333 (per sec)	4-4 to 4-9
INPUT:			
Single Inputs	Time: 1 — 3.0 ms Rate: 1000 — 333 (per sec)	2.7 — 4.7 ms 370 — 213 (per sec)	4-10 to 4-14
Single Ended Scan	Rate: 56 channels/second	50 channels/second	4-15 to 4-19
Double Ended Scan	Rate: 55 channels/second	35 channels/second	4-20

Chapter V
HIGH SPEED

5-1 The High Speed structure is for transfers that must take place at the fastest rates possible.

5-2 High Speed I/O operations take place in two steps. For output they are: 1) fill the buffer and 2) transfer the buffer to the output device. The two steps are reversed for input: 1) transfer data to the buffer from the input device and 2) empty the buffer. When calculating times for high speed transfers be sure to add the additional card time (table 5-1) to the software time.

5-3 OUTPUT USING THE FAST READ/WRITE BUFFER

5-4 To fill a buffer in the 9825A, the programmer writes to it as if he were writing to an external peripheral.

First the buffer is defined:
GPIO: buf "out",1000,2

This defines a fast read/write buffer named "out" to 1000 words.

HP-IB: buf "out",6000,3

This defines a fast read/write buffer named "out" to 6000 bytes. This is equivalent to 1000 16-bit duplex words.[1]

The defined buffer can now be filled by repeatedly writing to it.

GPIO: wtb "out", 20000 + A (1.5 ms)

HP-IB: (fmt3, "B", f4.0, "T", z)
wrt "out.3", A (2.6 ms)

The defined and filled buffer can be transferred out of the calculator as follows:

GPIO: tfr "out ", 9, 1000 (45 μs per word for high speed cards or using Automatic Handshake mode[2])

HP-IB: tfr "out", 723, 6000 (150 μs per word for high speed cards or using Automatic Handshake mode[2])

These transfer statements will transfer the contents of the buffer "out" to the 6940B. In this example, the card in slot 402 is addressed.

5-5 OTHER OUTPUT BUFFER TYPES

5-6 Interrupt Buffer. The previous example uses the fast read/write type of buffer. The interrupt type of buffer could also be used if the programmer desires to overlap the I/O operation with other computations. The fast read/write operation once begun, will pre-empt all other calculator operations until it is complete. The interrupt buffer, on the other hand interrupts other operations only when the peripheral device is ready for another date transfer.

5-7 Times to fill the interrupt buffer are the same as for the fast read/write buffer. Transfer times for the interrupt buffer are:

GPIO: tfr "out", 9, 1000 (.24 ms per word)
HP-IB: tfr "out", 723, 1000 (1.38 ms per equiv. 16 bit word)[1]

5-8 DMA Buffer. The DMA buffer allows data transfers to take place even faster than the fast read/write buffer without halting other computer operations. This form of transfer is only possible with 98032 opt 040 interfaces which have jumper 7 added. It is not possible using the HP-IB. Again, filling the buffer will proceed as before in the fast read/write example. The actual transfer speed will be faster than the fast read/write transfer speed even though other operations may take place concurrently. This is due to "cycle stealing" (refer to the 9825A Extended I/O Rom programming Manual for additional information).

GPIO: tfr "out", 9, 1000 (40 μs per word)

[1]An example of an equivalent 16 bit word is "0160T" or "A7777T"

[2]This mode is entered after a control word has been sent in which TME was off.

Table 5-1. Plug-In Cards

Model No.	Card Name	Increments to Automatic Handshake Speeds when using Medium and High Speed Structures in ms
HIGH SPEED CARDS		
69321B	D/A Voltage Converter	+.015
69322A	Quad DAC	+.065
69331A	Digital Output (gate-flag jumpered)	+0
69370A	D/A Current Converter	+.015
69422A	High Speed A/D	+0
69431A	Digital Input (gate-flag jumpered)	+0
LOWER SPEED CARDS		
69330A	Relay Output (gate-flag jumpered)	+7.0
69421A	Voltage Monitor	+6.4
69433A	Relay Output with Readback	+4.0
VARIABLE SPEED CARDS		
69335A	Stepping Motor Card	Time depends on the programmed data
69600B	Programmable Timer	

The 6940B will supply a 50 μs maximum, 40 μs typical, handshake cycle in automatic handshake mode (TME off).

5-9 Output High Speed Program Example.

```
0: "High Speed Output Example for GPIO":
1: buf "out",1000,2
2: moct;wtb 9,170160
3: for N=1 to 1000
4: wtb "out",50000+dtoN
5: next N;oni 9,"done"
6: tfr "out",9,1000
7: dsp A+1→A;gto +0
8: "done":dsp "Finished";end
*22622
```

Line 1 defines a fast read/write type buffer of 1000 words.

Line 2 sends a control word with TME, DTE, and SYE on (extended handshake mode).

The for-next loop on lines 3 to 5 loads up the buffer with an octal count from 1 to 1000.

Line 5 sets up interrupt linkage so that a branch to "done" will occur when the transfer is complete.

Line 6 initiates the transfer which proceeds at 45 μs (+ card time from table 5-1) per date transfer.

Line 7 is a count-and-display function.

When the transfer is complete the program displays "Finished".

5-10 INPUT

5-11 Due to the design of the 59500A interface, buffered input is not possible over the HP-IB. Buffered input is possible with GPIO and can be useful.

As with output, the buffer must first be defined:

GPIO: buf "in", 1000, 2

This defines a fast read/write buffer named "in" to 1000 words. The transfer is initiated after the appropriate control and address words have been sent by:

GPIO: tfr 9, "in", 1000 (42 μs per word)

5-12 Input Buffer Unloading

When the transfer is complete the data may be accessed by reading it from the buffer:

rdb ("in")→D [N] (1.4 ms per word)

Note that this buffered input is taking 1000 readings from one Multiprogrammer input card.

Scanned buffered input is not possible.

5-13 OTHER INPUT BUFFER TYPES

5-14 Buffered input over GPIO will also work using interrupt buffers (type 0) and DMA (type 4) buffers.

Interrupt buffer (type 0):

tfr 9, "in", 1000 (.24 ms per word)

DMA buffer (type 4):

(jumper 7 inserted on the 98032 opt 040)
tfr 9, "in", 1000 (30 μs per word)

The buffer unloading step is the same regardless of buffer type.

5-15 Input High Speed Program Example.

```
0: "High Speed Input Example for GPIO":
1: buf "in",1000,2
2: moct;wtb 9,170260;wti 0,11;wti 4,50000
3: oni 9,"done"
4: tfr 9,"in",1000
5: dsp A+1→A;gto +0
6: "done":fmt 1,f4.0,2x,f6.0
7: for N=1 to 1000
8: wrt 16.1,N,rdb("in")
9: next N
10: end
*7092
```

Line 1 defines a fast read/write type buffer of 1000 words.

Line 2 sends a control word setting up the input mode of the 6940B (ISL, TME, SYE) and puts card address 5 on the data lines.

Interrupt linkage to "done" sets up a branch to take place when the transfer is complete.

Line 4 begins the transfer from the 6940B to the buffer which proceeds at 42 μs per data transfer (+ card time from table 5-1).

Line 5 is a count-and-display routine.

The for-next loop of lines 7 to 9 prints out the buffer.

Table 5-2. High Speed Summary

Operation		16 Bit Duplex	HP-IB	Text Reference Paragraph
OUTPUT	Time:	50 μs (worst case) 40 μs (typ.) (DMA buffer)	150 μs (typ.) (fast read/write buffer)	5-3 to 5-8
	Rate:	25000 per sec (typ.)	6600 per sec (typ.)	
INPUT	Time:	50 μs (worst case) 30 μs (typ.) (DMA buffer)	not possible	5-10 to 5-14
	Rate:	33,333 per sec (typ.)		

Appendix A
HP-IB FORMAT CONSIDERATIONS

A-1 To send data to the 6940B over the HP-IB the minimum requirement is a simple 'wrt 723, "data" ' to send constant data or 'fmt1,c,f4.0, c; wrt 723.1, "address", V, "T" ' to send variable data. However, the transmission speed can be increased by certain formatting techniques.

A-2 By deleting carriage return and line feed after each transmission, 10 μs can be eliminated. To accomplish this a "z" is placed in the format statement: 'fmt 1, c, z'.

A-3 Furthermore, by placing the constants in the format statement the actual 'wrt' statement can execute faster: 'fmt 1, "0160TA", f4.0, "T", z; wrt 723.1, V'. By doing this the character string is prepared for transmission at a noncritical time early in the program. When it comes time for the transmission, the string is ready and can be quickly sent out. The 'wrt' statement with the characters in the format will execute 710 μs faster than when the characters are contained in the 'wrt' statement itself.

A-4 For maximum speed always be sure not to send optional characters, such as leading zeroes, to Multiprogrammer. Leading zeroes are those which precede non-zero numbers in the numerical portion of data and control words. For example, 'A0025T' should be sent as 'A25T' for maximum throughput.

A-5 The address portion of the Multiprogrammer word remains in the 59500A until a new address is sent. This means that an address character need not be sent twice as is often done when "stacking words" to coordinate input transfers. So, when the programmer wants to wait for an input card in slot B to complete he can use (fmt 1, c, z) 'wrt 723.1, "BTX" ', instead of 'wrt 723.1, "BTBX" '. This will save the time required to handshake an additional character across the HP-IB.

Appendix B
9825A PROGRAMMING CONSIDERATIONS

B-1 If speed is to be maximized, certain programming techniques can be used to increase the 9825A portion of the throughput. It must be noted that speed benefits come at the expense of other considerations. For example, if speed is maximized, ease of troubleshooting is often minimized.

B-2 Multiple Statements Per Line

B-3 By placing as many statements on each line as will fit (up to 80 characters), line overhead time can be saved. Every time two lines are used when the program code would fit on one line, approximately 50 μs are wasted. However, to save the 50 μs per line, the programmer will make it more difficult to step through the program for debugging purposes.

B-4 Even more time is saved by combining output (and input where possible) operations into one command as in 'wtb 9, 170040,40000' vs. 'wtb9,170040;wtb 9, 40000'. The former requires 1.6ms, while the latter requires 2.2ms.

B-5 For Next Loop Time:

The following is a general guide for estimating the time taken up by the for-next loop overhead time:

½ ms per iteration

+1.2 ms one time set up

B-6 Making Use of Multiprogrammer "Dead Time"

B-7 A key to effective and fast Multiprogrammer throughput is to make use of 6940B "dead time". This is the time when the slower Multiprogrammer operations are timing out; for example, while the relays close or the stepping motor turns. While the slow operations are finishing, new Multiprogrammer operations cannot be programmed. The 9825A, however, is free to do internal calculations or program other devices.

B-8 An example of effective use of "dead time" was shown in the analog input section of this analysis. In the analog voltage scanning operation it takes time (4ms) for the relays to open and close. The analog voltages can be converted to decimal while the relay contacts are opening. Note that doing the conversion at this point did not slow the total scanning rate at all.

B-9 For very slow operations, the 9825A interrupt system can be used to inform the system when the operation has completed. See the 6940B Multiprogrammer User's Guide for the HP 9825A Calculator for more details on programming interrupts from the 6940B.

Appendix C
HOW THE TIMES WERE MEASURED

C-1 The hardware set up used to measure the throughput times was similar to the Time Interval Measurement set up described in the 6940B Brochure. A 69601B Frequency Reference and a 69435A Pulse Counter were used in a 6940B Multiprogrammer mainframe as described in the brochure.

C-2 The 1kHz reference of the 69601B is wired to the count up input of the 69435A (see figure C-1). This gives us the ability to measure elapsed time in milliseconds. This elapsed time counter will continuously count up; thus to measure a time interval, the counter must be zeroed prior to the event and read immediately after the event.

As an example of using this test set up to time a routine, the following program will time "dto 1025 ⟶ A". Note this uses the 16 bit duplex interface.

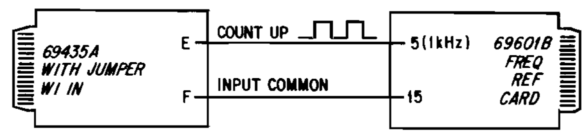

Figure C-1. Hardware Setup to Measure Execution Times.

C-3 The counter card is preset to a value less than zero, so that the elapsed time displayed will be only that of the "dto 1025⟶A" statement of line 3. By presetting to less than zero, the setup and the for-next loop times are subtracted out. In this case, the setup and for-next loop time is 53 ms.

```
0: "T-P ....Timing Outline.... GPIO":
1: dto(4096-53)→P
2: moct;wti 0,11;wtb 9,170040,100000+P
3: for N=1 to 1000;dto1025→A;next N
4: moct;wtb 9,170240;wti 4,100000;rdi 4→Z;dsp otdZ,"milliseconds"
5: end
*26789
```

HEWLETT hp PACKARD

For more information, call your local HP sales office or East (301) 948-6370 • Midwest (312) 255-9800 • South (404) 955-1500 • West (213) 877-1282; Or write, Hewlett-Packard, 1501 Page Mill Rd., Palo Alto, California 94304. **In Europe**, Hewlett-Packard S.A., 7, rue du Bois-du-Lan, P.O. Box, CH-1217 Meyrin 2-Geneva, Switzerland. **In Japan**, Yokogawa-Hewlett-Packard Ltd., 29-21, Takaido-Higashi 3-chome, Suginami-ku, Tokyo 168

5952-4017 SEPTEMBER 1978 10/12/78/R1 PRINTED IN U.S.A.