

Writing Drivers for Summagraphics® MM® Series Graphics Tablets

Introduction

This technical note was written to introduce concepts involved in writing a software driver for Summagraphics graphics tablets. Included are instructions on programming the tablet and decoding the tablet's binary output. The example is written in IBM® Basic version 3.0, but will run with most standard Basic interpreters with minor modifications to the statements configuring the communications port. The information covered is intended for programmers who have little experience writing device drivers. The experienced programmer who wants to go beyond the basic concepts is advised to consult one of the many books on advanced operating system programming.

The drivers and test programs have been tested on several IBM personal computers but are intended for illustration rather than economy of code. In addition to this technical note, you will need the technical reference manual for the Summagraphics MM Series tablet that you are using. Summagraphics sells several models that differ from the MM Series tablets in output format, so pay attention to the model and to the exact binary format used. Examples of other Summagraphics formats are outlined at the end of this note in the section *Writing Drivers for Bit Pad® and UIOF™ Formats*.

(IBM is a registered trademark of International Business Machines Corporation)
(Summagraphics, SummaSketch, MM, Bit Pad and MicroGrid are registered trademarks of Summagraphics Corp.)

Sample MM Program

This program contains the major elements involved in writing a simple device driver. Lines 40 and 50 open the port and configure it to accept the 9 bit 5 byte format used by the MM/SummaSketch and SummaSketch® Professional tablets. Lines 60-130 perform a self test, and lines 140-440 get information from the tablet and display it in a readable form.

```

10 ' Summagraphics MM Series Test and Driver Program (v1.1)
15 ' © 1987 Summagraphics Corp. Fairfield, CT.
20   CLEAR : CLS
30 ' ***** Open the Com port for the MM binary format *****
40   OPEN "COM1:9600,N,8,1,RS,CS,DS,CD" AS #1 ' Open port for MM format
50   OUT &H3FB,11 ' change to &H2FB,11 for COM2
60 ' ***** Self Test Routine *****
70   PRINT #1, "t": PRINT #1, "w";
80   BYTE10$ = INPUT$(1,#1)
90   IF ASC(BYTE10$) = 79 THEN GOTO 130 ELSE GOTO 100 ' 1812 in prox
100  IF ASC(BYTE10$) = 2 THEN GOTO 130 ELSE GOTO 110 ' 1812 out prox
110  IF ASC(BYTE10$) = 135 THEN GOTO 130 ELSE GOTO 120 ' 1201/961 out prox
120  IF ASC(BYTE10$) = 143 THEN GOTO 130 ELSE GOTO 450 ' 1201/961 in prox
130  PRINT "***** MM Tablet Passes self test *****": PRINT
140 ' ***** Driver Routine *****
150  PRINT #1, "B"; ' ** Point mode **
180  PRINT "Press Stylus or Cursor Button to Output Coordinates"
190  Print "Type -Ctrl Break- To Halt Program"
210 ' ***** Read binary input from tablet *****
220  BYTE1$ = INPUT$(1,#1)
230  IF ASC(BYTE1$) < 128 THEN GOTO 220
240  BYTE2$ = INPUT$(1,#1)
250  BYTE3$ = INPUT$(1,#1)
260  BYTE4$ = INPUT$(1,#1)
270  BYTE5$ = INPUT$(1,#1)
280 ' ***** Convert to ASCII *****

```

```

290  BYTE1 = ASC(BYTE1$)
300  BYTE2 = ASC(BYTE2$)
310  BYTE3 = ASC(BYTE3$)
320  BYTE4 = ASC(BYTE4$)
340  BYTE5 = ASC(BYTE5$)
350  IF (BYTE2>128) OR (BYTE3>128) OR (BYTE4>128) OR (BYTE5>128) THEN GOTO 220
360  *****Format the information for display *****
370  X = BYTE3*128+BYTE2
380  Y = BYTE5*128+BYTE4
390  FLAG = BYTE1 AND 7
400  PROX = BYTE1 AND 64: IF PROX =64 THEN PROXS = "OUT " ELSE PROXS = "IN "
410  ***** Display - Proximity, Button, X and Y - and messages for self test failure *****
420  PRINT PROXS; FLAG, X, Y
440  GOTO 220
450  PRINT " TABLET FAILS SELF TEST" : BEEP
460  PRINT "For more information consult the MM Technical Reference Manual"
470  END

```

Opening the Communications Port

Lines 40 and 50 open communication port #1 and configure it to except the MM binary data format (factory default). This statement opens COM1 at 9600 baud, no parity, 8 data bits and 1 stop bit. The remaining commands (RS,CS,DS,CD) disable the handshaking lines. At this point the attentive programmer will notice that the MM format is actually a 9 bit format (8 data, 1 parity). One limitation of IBM Basica is that the OPEN "COM... statement can except only 8 data bits including the parity bit. Since the MM format actually contains 9 bits, line 50 modifies the OPEN statement and allows the parity bit to be handled without trouble. OUT &H3FB,11 is used for COM1 and &H2FB,11 is used for COM2. For more information on opening the communications port see the *IBM Personal Computer Technical Reference* under Line Control Register. For computers other than IBM and 100% compatibles, the COM port can be set to the MM factory default settings of odd parity, 8 data bits and 1 stop bit.

Self Test

Output Format of Send Test Results ("w") For MM 1201, MM 961 and SummaSketch

Stop Bit	P	MSB							LSB	Start Bit
		7	6	5	4	3	2	1		
1	P	T	0	0	0	PR	D	C	A	0

Output Format of Send Test Results ("w") For MM 1812 and SummaSketch Professional

Stop Bit	P	MSB							LSB	Start Bit
		7	6	5	4	3	2	1		
1	P	0	T	0	0	PR	D	C	A	0

Definitions --

- A analog circuitry test; pass = 1, fail = 0
- C cursor/stylus connection and cursor/stylus coil operation test; pass = 1, fail = 0
- D digital circuitry test; pass = 1, fail = 0
- PR cursor/stylus on/off tablet; on = 1, off = 0

T total test result based on the results of tests A, C and D; pass = 1, fail = 0

Lines 70 - 130 ask the tablet to report the results of a self test. Line 70 sends the tablet the character "t" to perform the test, followed by the character "w" to report the results of the test in a single byte. Line 80 reads the test result (see tables above). Lines 90 - 120 convert the binary report (BYTE10\$) to its ASCII decimal equivalent, and test the result. Depending on the result, the program jumps to line 130 or to line 450 to display the appropriate message.

Driver

MM Packed Binary Report Format

Stop Bit	P	MSB 7	6	5	4	3	2	1	LSB 0	Start Bit	Byte Number
1	P	PH	PR	T	Sx	Sy	Fc	Fb	Fa	0	#1
1	P	0	X6	X5	X4	X3	X2	X1	X0	0	#2
1	P	0	X13	X12	X11	X10	X9	X8	X7	0	#3
1	P	0	Y6	Y5	Y4	Y3	Y2	Y1	Y0	0	#4
1	P	0	Y13	Y12	Y11	Y10	Y9	Y8	Y7	0	#5

Definitions --

LSB	least significant bit
MSB	most significant bit
F	flag bit identifying the stylus or cursor button number (see MM Tech. Ref. for details)
Sy or Sx	sign bit for X or Y: 1 is positive, 0 is negative
T	tablet identifier, choice of 0 or 1
PR	proximity: 0 is "in" proximity, 1 is "out" of proximity
PH	phasing bit, always 1
P	parity bit
X0 to X13	X coordinate
Y0 to Y13	Y coordinate

The driver routine runs from line 150 to line 440 and is written in four parts. Line 150 sends the tablet the character "B" that puts the tablet into "point mode" where coordinates are reported only when a stylus/cursor button is depressed. All control of the tablet's mode should occur in this line. For example, if you want the tablet to operate in "stream mode", where coordinates are issued continuously, you will first need to send a command to slow down the report rate, and then send the "stream mode" command. Line 150 would read -- 150 PRINT #1, "T": PRINT #1, "@": In compiled programs that run faster than interpreted Basica, there is usually no need to slow the rate to use "stream mode".

Lines 220 - 270 read and synchronize the output from the tablet. Line 230 tests to see if the byte just examined was the first, if the phasing bit is not detected it tries again until the phasing bit, identifying the first byte, is detected.

Lines 280 - 340 convert the binary input to their ASCII decimal equivalents and line 350 tests to see that each byte is within the range of possible values. If any byte is out of range, the program loops to line 220 and new input is gathered from the tablet. Note that BYTE1\$ is not tested because the phasing bit causes the value to be greater than 127 (see line 230 that uses this value as a test for BYTE1).

Lines 370 - 400 put the information into a readable form. Lines 370 and 380 assemble the information from bytes 2 and 3 into the X coordinate and likewise assemble the information from bytes 4 and 5 into the Y coordinate. Line 390 extracts from BYTE1 information about the buttons. This is done by "masking" the bits in BYTE1 not related to buttons. Using the logical AND operator, BYTE1 AND 00000111 (binary 7 = 00000111) will cause any bits in BYTE1 except bits 0, 1 and 2 to equal 0 (see table below).

Logical AND

X	Y	X AND Y
1	1	1
1	0	0
0	1	0
0	0	0

Likewise, line 400 decodes the proximity bit by masking BYTE1 with 64 (binary 64 = 01000000).

Here is an example of how the binary masks are used to decode button information (FLAG) --

- Assume BYTE1 = 10000100. Such a value would result from pressing button 4 while the cursor is in proximity (remember that the first bit, the phasing bit, is always = 1).
- Expressing line 390 in binary -- FLAG = 10000100 AND 00000111. The mask (00000111) causes the phasing bit to be set to 0, leaving only the information about the buttons.
- Therefore FLAG = 00000100 = 4, and the program reports that button 4 was depressed.

Lines 420 - 440 simply format the data for display and cause the program to loop to line 220 to gather another coordinate.

Writing a Diagnostic Self Test

Using masks and logical operators, you can write a diagnostic program that will interpret the self test and print the results.

```

10 ***** Diagnostics for MM 1201/961 *****
20 OPEN "COM1:9600,N,8,1" AS #1
30 OUT &H3FB,11
40 PRINT #1, "t": PRINT #1, "w";
50 BYTE10$ = INPUT$(1,#1)
60 BYTE10 = ASC(BYTE10$)
100 ANALOG = BYTE10 AND 1: IF ANALOG = 1 THEN 500 ELSE 600
110 CURSOR = BYTE10 AND 2: IF CURSOR = 2 THEN 510 ELSE 610
120 DIGITAL = BYTE10 AND 4: IF DIGITAL = 4 THEN 520 ELSE 620
500 PRINT "Analog circuitry test - PASS": GOTO 110
510 PRINT "Cursor/Stylus connection and operation test - PASS":GOTO 120
520 PRINT "Digital circuitry test - PASS":GOTO 1000
600 PRINT "Analog circuitry test - FAIL": GOTO 110
610 PRINT "Cursor/Stylus connection and operation test - FAIL": GOTO 120
620 PRINT "Digital circuitry test - FAIL"
1000 END

```

Writing Drivers for BitPad and UIOF Formats

The binary output format is the major difference between models of Summagraphics graphics tablets, but other important differences exist that will influence your program. For example, the MicroGrid® Series tablets contain a richer variety of control commands because of the MicroGrids' expanded capabilities. Therefore, attention must be paid, not only to the binary format, but also to differences in control commands, switch settings and in basic tablet operation. Always consult the technical reference of the tablet series before you begin to code your driver.

The programs below are for reference only and the Self Test routine outlined in the Sample MM Driver WILL NOT WORK with other tablet models. Consult the appropriate technical reference for Self Test details.

Bit Pad Format

Using the same notation as before, except that SB = Stop Bits, the Bit Pad format can be expressed in the following form. SB is used in the Bit Pad format because these parameters can be set to equal 1 or 2 by changing DIP switches. There are a few differences between Summagraphics Bit Pad One and Bit Pad Two tablets. Among the important differences are Point Mode and proximity. Point Mode is set by switches in the Bit Pad One, but is remotely controlled in the Bit Pad Two by sending the ASCII character "P". Be sure to use Point Mode, otherwise the display cannot keep up with the tablet and a buffer overflow will occur. The format differs between the Bit Pad One and the Bit Pad Two in bit 0 byte 1 where the Bit Pad Two loads the proximity bit (PR). This bit is 0 in the Bit Pad One.

Bit Pad Packed Binary Report Format

Stop Bit7	MSB 7	6	5	4	3	2	1	LSB 0	Start Bit	Byte Number
SB	P	PH	0	0	Fb	Fa	0	PR	0	#1
SB	P	0	X5	X4	X3	X2	X1	X0	0	#2
SB	P	0	X11	X10	X9	X8	X7	X6	0	#3
SB	P	0	Y5	Y4	Y3	Y2	Y1	Y0	0	#4
SB	P	0	Y11	Y10	Y9	Y8	Y7	Y6	0	#5

Sample Bit Pad Driver

```

10 OPEN "COM1:9600,E,7,2" AS #1
30 BYTE1$ = INPUT$(1,#1)
40 IF ASC(BYTE1$) < 64 THEN GOTO 30
50 BYTE2$ = INPUT$(1,#1)
60 BYTE3$ = INPUT$(1,#1)
70 BYTE4$ = INPUT$(1,#1)
80 BYTE5$ = INPUT$(1,#1)
90 BYTE1 = ASC(BYTE1$)
100 BYTE2 = ASC(BYTE2$)
110 BYTE3 = ASC(BYTE3$)
120 BYTE4 = ASC(BYTE4$)
140 BYTE5 = ASC(BYTE5$)
150 IF (BYTE2>64) OR (BYTE3>64) OR (BYTE4>64) OR (BYTE5>64) THEN GOTO 30
160 X = BYTE3*64+BYTE2
170 Y = BYTE5*64+BYTE4
180 FLAG = BYTE1 AND 12
190 PROX = BYTE1 AND 1: IF PROX = 1 THEN PROX$ = "OUT " ELSE PROX$ = "IN "
200 PRINT PROX$; FLAG, X, Y
210 GOTO 30

```

UIOF Format

Be sure to set the DIP switches for Point Mode, 2 stop bits and even parity.

UIOF Packed Binary Report Format

Stop Bit7	MSB 7	6	5	4	3	2	1	LSB 0	Start Bit	Byte Number
SB	P	PH	0	0	0	0	T	PR	0	#1
SB	P	0	0	Fe	Fd	Fc	Fb	Fa	0	#2
SB	P	0	X5	X4	X3	X2	X1	X0	0	#3
SB	P	0	X11	X10	X9	X8	X7	X6	0	#4
SB	P	0	0	Sx	X15	X14	X13	X12	0	#5
SB	P	0	Y5	Y4	Y3	Y2	Y1	Y0	0	#6
SB	P	0	Y11	Y10	Y9	Y8	Y7	Y6	0	#7
SB	P	0	0	Sy	Y15	Y14	Y13	Y12	0	#8

Sample UIOF Driver

```

10 OPEN "COM1:9600,E,7,2" AS #1
30 BYTE1$ = INPUT$(1,#1)
40 IF ASC(BYTE1$) < 64 THEN GOTO 30
50 BYTE2$ = INPUT$(1,#1)
60 BYTE3$ = INPUT$(1,#1)
70 BYTE4$ = INPUT$(1,#1)
80 BYTE5$ = INPUT$(1,#1)
90 BYTE6$ = INPUT$(1,#1)
100 BYTE7$ = INPUT$(1,#1)
110 BYTE8$ = INPUT$(1,#1)
120 BYTE1 = ASC(BYTE1$)
130 BYTE2 = ASC(BYTE2$)
140 BYTE3 = ASC(BYTE3$)
150 BYTE4 = ASC(BYTE4$)
160 BYTE5 = ASC(BYTE5$)
170 BYTE6 = ASC(BYTE6$)
180 BYTE7 = ASC(BYTE7$)
190 BYTE8 = ASC(BYTE8$)
200 IF (BYTE2>64) OR (BYTE3>64) OR (BYTE4>64) OR (BYTE5>64) THEN GOTO 30
210 IF (BYTE6>64) OR (BYTE7>64) OR (BYTE8>64) THEN GOTO 30
220 X = BYTE5*4096+BYTE4*64+BYTE3
230 Y = BYTE8*4096+BYTE7*64+BYTE6
240 FLAG = BYTE2 AND 31
250 PROX = BYTE1 AND 1: IF PROX = 1 THEN PROXS = "OUT " ELSE PROXS = "IN "
260 PRINT PROXS; FLAG, X, Y
270 GOTO 30

```