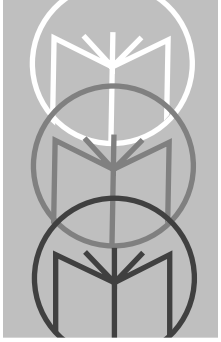


Chapter 6 Handwriting Recognition

Introduction

The PPT 41XX terminal uses the Communication Intelligence Corporation (CIC) Handwriter Recognition System (HRS) to process a wide variety of handwritten characters and gestures drawn on the terminal screen. HRS accepts handwritten input in the form of strokes with the pen on the screen, is able to interpret a wide variety of styles, and returns recognized results to the calling application. Character recognition software converts the pen strokes that it recognizes to ASCII text.

Handwriting recognition for the PPT 41XX is available in two versions:

- *Numeric Only*

  This is the version shipped with the PPT 41XX Software Development Kit (SDK). It consists of the following files:

  - LOADC5.EXE (the handwriting recognition TSR)
  - GROUP000.BIN, GROUP001.BIN, GROUP002.BIN, and GROUP003.BIN (used for numeric recognition)

- *Alphanumeric*

  This version is available for special purchase. It consists of the following files:

  - LOADC5.EXE (the handwriting recognition TSR)
  - Fourteen .BIN files with names GROUP000 through GROUP013 (used for numeric, alphabetic, and punctuation recognition)

The numeric-only version features recognition speed and lower memory requirements; the alphanumeric features wider recognition capabilities at the price of greater memory. The relative memory requirements are given in the following chart:

Version	**Conventional Memory Required (Kbytes)**	**Expanded Memory Required (Kbytes)**
Numeric Only	38	131
Alphanumeric	70	384

The HRS is usually loaded as a TSR from the AUTOEXEC.BAT file when the terminal is booted. If the appropriate statements have not been placed in the AUTOEXEC.BAT file or if you choose not to use this method of invoking the HRS, you can change to the directory in which LOADC5.EXE and the associated GROUP0*xx*.BIN files are stored on the terminal and enter:

LOADC5

on the command line. This loads the HRS as a TSR program and makes available to application programs the API described in *Supported API Commands*.

Handwriter Recognition System (HRS) API

General Operating Features

This API provides access to HRS software by application developers. Application programs access the HRS API via **INT 0x15**, specification of function code **0xC5** in the AH register, and the desired handwriting recognition service with the appropriate subfunction code in register AL. Depending on the service requested, the program may also have to identify in other registers or register pairs the addresses of data structures, recognition and character segmentation modes, position coordinates, size parameters, etc.

The application must perform inking and erasing of gestures and recognition characters. Calculating the location of a character must also be performed manually by looking at the strokes comprising the character. The application must convert from virtual mouse screen coordinates to the HRS virtual tablet coordinate system.

Pen Input with the HRS API

An application running with the HRS API receives pen input through a set of functions which emulate the interrupt calls of the Microsoft Mouse Driver. Refer to the *Supported API Commands* section of *Chapter 5, Mouse Emulator* for a list and descriptions of supported mouse emulator functions accessible on the PPT 41XX.

The status of the pen tip emulates the status of the left mouse button. The pen down state translates to left mouse button pressed; the pen up state translates to left mouse button released. When an application receives pen strokes from the user for recognition, all input points must be received. In these instances, the pen input event loop should be interrupt-driven through the following mouse emulator service:

INT 0x33, Function0x0C (**Set User-Defined Pen Event Handler**)

This mouse emulator function is described in detail in the *Supported API Commands* section of *Chapter 5, Mouse Emulator*.

Recognition under the HRS API

Inking must be performed by the application, and the application must maintain arrays of pen points comprising strokes and send these strokes to the HRS through an interrupt call. An interrupt call also obtains recognition results. One character is returned per call, along with the application-assigned ID numbers of the strokes that comprise that character. The same process is used for the recognition of both characters and gestures. Only the symbol set to be recognized is changed.

Stroke Input and Recognition Output

A “complete stroke” is the sequence of pen-down points occurring between two pen-up points, regardless of the shape of the mark produced.

The caller passes complete strokes to the recognition software through an HRS API function call, one stroke per call. The HRS buffers these strokes until a character is complete, then recognizes the character and places the result in an output queue. The calling application retrieves these results one at a time through another API function call, one result per call. The application should provide a unique stroke identifier (UID) for each stroke, and the HRS reports the associated UIDs with its recognition results.

Recognition Modes

HRS software on the PPT 41XX enables the calling program to select one of the following modes of recognition:

- *Numeric only*
  The application program identifies this recognition mode by calling the **Set Symbol Set** service (**Subfunction 0x00**) and passing 0x01 in the BH register.
- *Alphanumeric and punctuation*
  The application program identifies this recognition mode by calling the **Set Symbol Set** service (**Subfunction 0x00**) and passing 0x04 in the BH register.

  Note: This recognition mode is available only with the alphanumeric version of the HRS.

Character Segmentation Modes

In calls to the HRS application programs must specify one of the following character segmentation modes:

- *Gridded Mode*
  In gridded mode, the writing surface is divided into a grid of boxes of uniform size. The calling program specifies the origin of the grid and the width and height of its boxes.

  The application program identifies this character segmentation mode by calling the **Set Symbol Segmentation Mode** service (**Subfunction 0x04**) and passing 0x02 in the BH register. It passes size and location parameters in calls to **Subfunctions 0x09**, **0x0A**, **0x0B**, and **0x0C**. (See *Supported API Commands*.)
- *Lined mode*
  In lined mode, the writing surface is divided into uniformly spaced horizontal

lines on which the user writes. The calling program specifies the origin of the set of lines and their vertical separation.

The application program identifies this character segmentation mode by calling the **Set Symbol Segmentation Mode** service (**Subfunction 0x04**) and passing 0x03 in the BH register. It specifies the origin of the grid and the vertical spacing between its lines in calls to HRS **Subfunctions 0x09**, **0x0A**, and **0x0B**. (See *Supported API Commands*.)

Note: In both gridded and lined modes, each box or each space between lines is further divided into subsections that may be used for spatial identification of commas, periods, and other position-dependent symbols.

- *Gridless mode*
  In gridless mode the calling program specifies only the origin of the text entry region. HRS uses spatial criteria to perform segmentation.

  The application program identifies this character segmentation mode by calling the **Set Symbol Segmentation Mode** service (**Subfunction 0x04**) and passing 0x01 in the BH register. It specifies the origin of the grid in calls to HRS **Subfunctions 0x09** and **0x0A**. (See *Supported API Commands*.)

  Note: In all three character segmentation modes, the application can force immediate recognition of all unrecognized strokes pending by calling HRS **Subfunction 0x03** (see **Force Recognition of Strokes** in *Supported API Commands*). In lined and gridless modes, HRS uses spatial criteria to segment the pending strokes into characters, if necessary.

The following are the recognition details of the spatial criteria used in HRS:

- The *period* (.), the *comma* (,), and the *underscore* (_) are written in the *lower 35%* of the box.
- The *tilde* (~) and the *single quote* (') are written in the *upper 43%* of the box.
- The *double quote* (") is written in the *upper 35%* of the box.

Tablet versus Mouse Coordinate Systems

The HRS operates in a virtual tablet coordinate system. Therefore all points passed to it must be in tablet units. Tablet points are calculated from the bottom left corner of the tablet. The HRS is optimized for tablet units spaced in increments of 0.1 mm. Mouse points are calculated from the top left corner of the screen, and are measured in virtual screen pixels.

Forced Segmentation

HRS software can perform segmentation only by using spatial criteria from the points passed to it. However, characters should be separated also through the use of temporal criteria. For example, if the pen is lifted from the pad for a specified amount of time, the application assumes that the last written character is completed. This is called a timeout. If the user starts writing in a different writing area, draws a gesture, or executes a command, the application assumes that the current character is completed. To allow the application to specify segmentation at these points, the HRS API provides a command that forces segmentation and recognition of all pending strokes. See **Force Recognition of Strokes** (**INT 15H, Function C5H, Subfunction 03H**) in *Supported API Commands*.

Character Recognition

Figure 6-1 illustrates how the program might use the HRS API to perform character recognition.

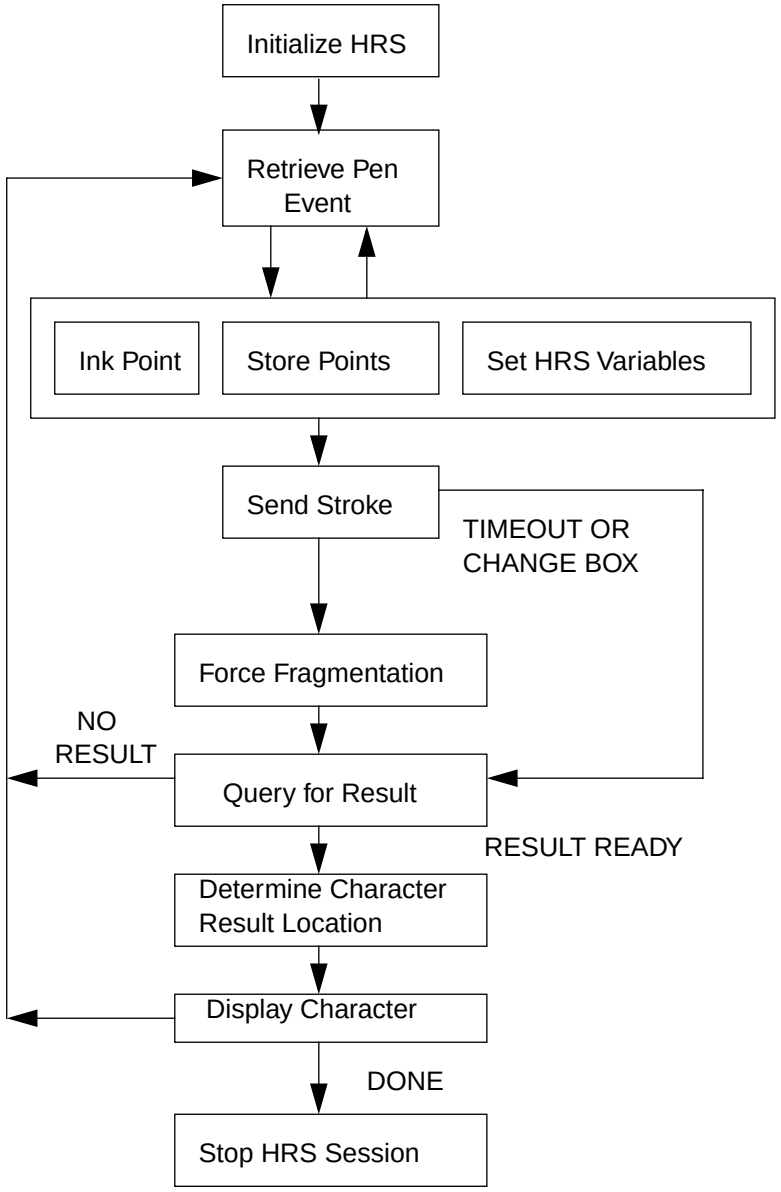


Figure 6-1. Handwriter Recognition System API Use

Gesture Recognition

The application must determine when a symbol written is a gesture by context (e.g., location at which the gesture is written) or mode (e.g., currently selected state). If the strokes written constitute a gesture, the application should perform the following functions:

- set the recognition mode to gridless
- set the Recognizer symbol set to the gesture symbol set
- call the Recognizer to submit the stroke, segment the symbol, and retrieve the result as for any character

Supported API Commands

All HRS API calls are made via software interrupt 0x15. The interrupt must be called with 0xC5 in the AH register and the subfunction code for the desired recognition service in the AL register. Refer to Table 6-1 for subfunction codes that are valid in the PPT 41XX implementation of the HRS and their associated services.

Pointers to data structures, when needed, are passed in the DX:DI register pair. The HRS API returns a status code in AX and single-word results in CX. It returns pointers to any structure in DX:DI.

This section consists of the following subsections:

- *List of HRS API Commands* lists the API commands supported by the handwriting recognition software on PPT 41XX terminals.
- *Command Descriptions* provides individual descriptions of the API commands (functions) supported by the handwriting recognition software.

List of HRS API Commands

Table 6-1 lists the functions supported by the recognition software on PPT 41XX terminals. The list is sorted by the subfunction code that an application must assign to the AL register with **AH = 0xC5** prior to calling **INT 0x15** to request the desired service. These functions are described in *Command Descriptions*.

Table 6-1. HRS API Commands (INT 0x15)

Function Code	Subfunction Code	Handwriter Recognition System Service Name
0xC5	0x00	*Set Character/Symbol Set*
0xC5	0x01	*Get Recognition Result*
0xC5	0x02	*Submit Stroke Information*
0xC5	0x03	*Force Recognition of Strokes*
0xC5	0x04	*Set Symbol Segmentation Mode*
0xC5	0x05	*Set Automatic Space Option*
0xC5	0x06	*Set Automatic Newline Option*
0xC5	0x09	*Set X-Origin*
0xC5	0x0A	*Set Y-Origin*
0xC5	0x0B	*Set Gridded Cell Width*
0xC5	0x0C	*Set Gridded and Lined Cell Height*
0xC5	0x0D	*Discard Strokes and Recognition Results*
0xC5	0x27	*Set Recognition Result Mode*

Command Descriptions

The following descriptions of the Handwriting Recognition API services listed in Table 6-1 are given in subfunction code order.

Set Character/Symbol Set

Function: 0xC5

Subfunction: 0x00

Description

Sets the HRS to recognize the symbol set passed in the BH register.

Interrupt

0x15

Input Registers

AH = 0xC5

AL = 0x00

BH = Symbol set for HRS to recognize as follows:

0x01: Numeric only
0x03: Alphabetic only
0x04: Alphanumeric
0x06: PenDOS gestures

Note: The numeric-only HRS software shipped with the PPT 4100/4110/4140 SDK supports only the mode specified by BH = 0x01 (Numeric only). The alphanumeric version is a special purchase option and supports all modes.

Output Registers

AX = Call completion status as follows:

0x00: No error
0x01: Invalid subfunction code

Get Recognition Result

Function: 0xC5

Subfunction: 0x01

Description

Fills the data structure (CICQUERY) pointed to by the DX:DI register pair with information about the next recognized character in the HRS output queue, removes the recognition result from the output queue, and returns to the calling application.

Interrupt

0x15

Input Registers

AH = 0xC5

AL = 0x01

DX:DI = Address of data structure CICQUERY, defined as follows:

```
CICQUERY     STRUC

   Result    DW  0   ; 0x7FFFF = Not recognized
   NumStks   DW  0   ; Number of strokes in character
   UID1      DW  0
   UID2      DW  0
   UID3      DW  0
   UID4      DW  0
   UID5      DW  0
   UID6      DW  0
   UID7      DW  0
   UID8      DW  0
   UID9      DW  0
   UID10     DW  0
   UID11     DW  0
   UID12     DW  0
CICQUERY  ENDS
```

The following are descriptions of the information returned in this data structure:

Result
The value returned in this field depends on whether the HRS recognition software has been set to two-byte mode or to one-byte ASCII mode. See HRS **Subfunction 0x27** (**Set Recognition Result Mode**). In two-byte mode, the result returned is the two-byte code of the character. In one-byte mode, 0 is returned in the high byte and the ASCII code of the recognized character in the low byte. The **Result** field in CICQUERY contains 0x7FFFF if the strokes were not recognized.

NumStks
This field contains the number of strokes in the recognized character. If there are no recognition reports in the output queue, zero is returned in **NumStks** and the other fields in the structure are undefined.

UID1 ...
These fields contain the stroke identifiers of the strokes that form the character. The stroke identifier numbers are assigned by the application with HRS **Subfunction 0x02** (**Submit Stroke Information**). The reserved value 0xFFFF is returned as the identifier for a space generated by the gridless or lined mode autospace feature, and the reserved value 0xFFFE is returned as the identifier for a newline generated by the gridless or lined mode auto-newline feature.

Output Registers

AX = Call completion status as follows:

0x00: No error
0x01: Invalid subfunction code

Submit Stroke Information

Function: 0xC5

Subfunction: 0x02

Description

Enables an application to assign a stroke identifier number to the **UID** field and the number of coordinate pairs that form a stroke in the **NumCoordPairs** field in the CICSTROKE data structure pointed to by the address that is passed by the application in the DX:DI register pair.

The value assigned to the **UID** field enables the application to identify the strokes comprising a recognition result.

The application stores the tablet points that compose the stroke in an array of CICCOORDPAIR structures, defined below. The maximum number of tablet coordinate pairs allowed per stroke is defined as **MAXCOORDPAIRS**. The x- and y-coordinates must be between 0 and 3048. The application typically must transform the screen coordinates of the pen input points into virtual tablet coordinates.

As each stroke is submitted, the HRS attempts to segment the pending strokes. Any recognition results are placed into the output queue.

Interrupt

0x15

Input Registers

AH = 0xC5

AL = 0x02

DX:DI = Address of data structure CICSTROKE, which is defined as follows:

```
CICSTROKE       STRUC
UID             DW      0     ; STROKE IDENTIFIER
NUMCOORDPAIRS   DW      0     ; NUMBER OF POINTS IN A STROKE
CICSTROKE       ENDS
```

The following are descriptions of the information provided by the application for this data structure:

UID
The stroke identifier number provided by the application to identify the strokes that comprise a recognition result.

NumCoordPairs
The number of coordinate pairs specified by the application as forming a stroke.

The following is the structure definition (CICCOORDPAIR) for coordinate pairs that immediately follow the CICSTROKE structure.

```
CICCOORDPAIR        STRUC
ABSX        DW      0       ; BETWEEN 0 AND 3048
ABSY        DW      0       ; BETWEEN 0 AND 3048
CICCORDPAIR         ENDS
```

Output Registers

AX = Call completion status as follows:
0x00: No error
0x01: Invalid subfunction code

Force Recognition of Strokes

Function: 0xC5

Subfunction: 0x03

Description

Segments any pending strokes and queues recognition results before returning.

Any strokes that have been submitted but for which recognition results have not yet been produced are “forced” to recognize, if possible. For example, a single vertical stroke results in the recognition of a numeral “1” or a lower case “l”, depending on the recognition mode, as opposed to being the first stroke in a multi-stroke character such as “B”, “D”, “E”, etc.

Interrupt

0x15

Input Registers

AH = 0xC5

AL = 0x03

Output Registers

AX = Call completion status as follows:

0x00: No error
0x01: Invalid subfunction code

Set Symbol Segmentation Mode

Function: 0xC5

Subfunction: 0x04

Description

Sets the HRS into one of three possible symbol segmentation modes:

- Gridless
- Gridded (Boxed)
- Lined

These modes determine how the HRS segments symbols and the level of recognition accuracy. The *gridded (boxed)* mode provides the highest level of recognition accuracy; the *lined* mode provides a medium level of recognition accuracy; the *gridless* mode provides the lowest level of recognition accuracy. See **Note**.

Interrupt

0x15

Input Registers

AH = 0xC5

AL = 0x04

BH = Symbol segmentation mode as follows:

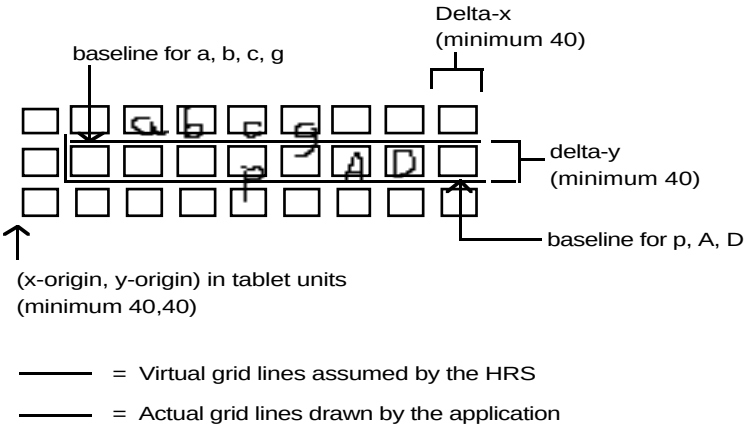
0x01: Gridless
0x02: Gridded (Boxed)
0x03: Line

Output Registers

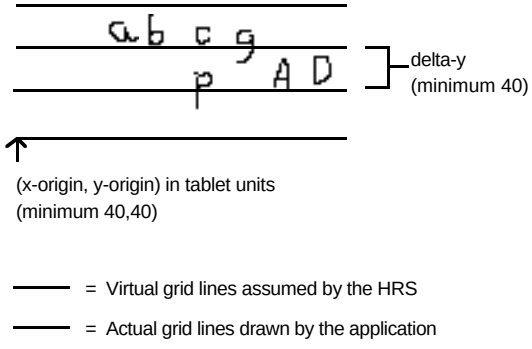
AX = Call completion status as follows:

0x00: No error
0x01: Invalid subfunction code

The following illustrations depict the Recognizer variables that must be set for boxed or line mode.



Note: In boxed mode, extra room is provided outside the grid for overwritten characters and descenders.



The user interface can be designed in any manner regardless of which symbol segmentation mode is used. For example, the input fields can be displayed as grid boxes but with the symbol segmentation mode set to gridless or lined. The recommendation is to use the boxed mode where possible, use lined mode if boxed mode is not practical, and avoid use of the gridless mode.

Set Automatic Space Option

Function: 0xC5

Subfunction: 0x05

Description

Enables an option that places a space recognition result in the recognition result queue if there is a space left between written symbols.

Even if the space between the written characters appears to be more than one character wide, only one space recognition result is placed in the recognition result queue.

This option takes effect only in lined and gridless modes. When a space recognition result is returned by **Subfunction 0x01** (**Get Recognition Result**), the stroke identifier (**UID1, ..., UID12**) and number of strokes (**NumStks**) fields in the CICQUERY data structure is set to 0xFFFE and 0x0001, respectively. See the description of the **Get Recognition Result** service for the definitions of the CICQUERY structure and its fields.

Interrupt

0x15

Input Registers

AH = 0xC5

AL = 0x05

Output Registers

AX = Call completion status as follows:

0x00: No error
0x01: Invalid subfunction code

Set Automatic Newline Option

Function: 0xC5

Subfunction: 0x06

Description

Enables an option that places a newline (0x0A) result in the recognition result queue if the stroke being processed begins on a line different from the one on which the previous stroke was made.

Even if the number of lines between the written symbols is more than one line, only one newline recognition result is placed in the recognition result queue.

This option takes effect only in lined mode. When a newline recognition result is returned by **Subfunction 0x01** (**Get Recognition Result**), the stroke identifier (**UID1, ..., UID12**) and number of strokes (**NumStks**) fields in the CICQUERY data structure is set to 0xFFFE and 0x0001, respectively. See the description of the **Get Recognition Result** service for the definitions of the CICQUERY structure and its fields.

Unlike all other symbols, the newline recognition result will be 0x000A regardless of whether HRS **Subfunction 0x27** (**Set Recognition Result Mode**) is set to ASCII or two-byte recognition results.

Interrupt

0x15

Input Registers

AH = 0xC5

AL = 0x06

Output Registers

AX = Call completion status as follows:

0x00: No error
0x01: Invalid subfunction code

Set X-Origin

Function: 0xC5

Subfunction: 0x09

Description

Used with **Subfunction 0x0A** (**Set Y-Origin**) to specify the lower left corner of the recognition area in the virtual tablet coordinate system.

Interrupt

0x15

Input Registers

AH = 0xC5

AL = 0x09

CX = X-origin in tablet units (See *Tablet versus Mouse Coordinate Systems*.)

Output Registers

AX = Call completion status as follows:

0x00: No error
0x01: Invalid subfunction code

Set Y-Origin

Function: 0xC5

Subfunction: 0x0A

Description

Used with **Subfunction 0x09 (Set X-Origin)** to specify the lower-left corner of the recognition area in the virtual tablet coordinate system.

Interrupt

0x15

Input Registers

AH = 0xC5

AL = 0x0A

CX = Y-origin in tablet units (See *Tablet versus Mouse Coordinate Systems*.)

Output Registers

AX = Call completion status as follows:

0x00: No error
0x01: Invalid subfunction code

Set Gridded Cell Width

Function: 0xC5

Subfunction: 0x0B

Description

Sets the width of each grid box.

This subfunction has an effect only when the symbol segmentation mode is gridded (boxed). See **Subfunction 0x04** (**Set Symbol Segmentation Mode**) for the HRS service used to set this mode.

Interrupt

0x15

Input Registers

AH = 0xC5

AL = 0x0B

CX = width of box in tablet units (See *Tablet versus Mouse Coordinate Systems*.)

Output Registers

AX = Call completion status as follows:

0x00: No error
0x01: Invalid subfunction code

Set Gridded and Lined Cell Height

Function: 0xC5

Subfunction: 0x0C

Description

Sets the height of grid boxes in gridded mode or the distance between lines in lined mode.

This subfunction has an effect only when the symbol segmentation mode is gridded (boxed) or lined. See **Subfunction 0x04** (**Set Symbol Segmentation Mode**) for the HRS service used to set this mode.

Interrupt

0x15

Input Registers

AH = 0xC5

AL = 0x0C

CX = height of box or distance between lines in tablet units (See *Tablet versus Mouse Coordinate Systems*.)

Output Registers

AX = Call completion status as follows:

0x00: No error
0x01: Invalid subfunction code

Discard Strokes and Recognition Results

Function: 0xC5

Subfunction: 0x0D

Description

Discards any submitted strokes not yet recognized as a symbol and any recognition results not yet obtained by the calling application via HRS **Subfunction 0x01** (**Get Recognition Result**).

This service does not reset any other parameters.

Interrupt

0x15

Input Registers

AH = 0xC5

AL = 0x0D

Output Registers

AX = Call completion status as follows:

0x00: No error
0x01: Invalid subfunction code

Set Recognition Result Mode

Function: 0xC5

Subfunction: 0x27

Description

Sets the HRS recognition result mode to return recognition results in either one byte ASCII code or two byte code.

Interrupt

0x15

Input Registers

AH = 0xC5

AL = 0x27

BH = Recognition result mode option as follows:

0x00: One byte ASCII recognition results
0x01: Two byte recognition results

Output Registers

AX = Call completion status as follows:

0x00: No error
0x01: Invalid subfunction code