Honeywell

OEM API OCR Programming

User's Guide

Disclaimer

Web Address: www.honeywellaidc.com

Table of Contents

OCR Programming

The following instructions are for programming your scan engine for optical character recognition (OCR).

Currently there are three functions in the Decoder API used for configuring the decoder for reading OCR.

OCR API Functions

- decSetOCRMode (see Enabling OCR Reading on page 2)
- decSetOCRTemplates (see Pre-Defined OCR Templates on page 2)
- decSetOCRUserTemplate (see Custom OCR Templates on page 5)

OCR Data Type Definitions

Throughout this document, the following variable type is used. This data type is defined in the Decoder_OEM.h header file.

OCREnable_t

This enum defines the value of the OCR settings used in conjunction with decSetOCRUserTemplate and decOCRMode.

```
typedef enum{
   OCR_OFF = 0,
   OCR_NORMAL_VIDEO,
   OCR_INVERSE,
   OCR_BOTH
}  OCREnable_t;
```

The scan engine will read OCR-A, OCR-B, MICR E-13B, and SEMI Font in a 6 to 60 point OCR typeface. You can either select a pre-defined OCR template, or create your own custom template for the type of OCR format you intend to read.

The following OCR characters are currently supported:

OCR-A:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

0123456789

#$&()*+-./<>@\€£¥

OCR-B:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

0123456789

#$&()*+-./<>@\€£¥

MICR E-13B:

0123456789⑆⑇⑈⑉

Enabling OCR Reading

Result_t decSetOCRMode (

OCREnable_t nEnable

);

The decSetOCRMode function is used to ONLY enable\set OCR mode. This function should be used when only the mode needs to be configured\changed. The nEnable parameter can be one of the values from the OCREnable_t structure.

Note: Once OCR reading is enabled, you must select a Pre-Defined Template, page 2, or create a Custom OCR Template, page 5, in order to read OCR characters.

Pre-Defined OCR Templates

Result_t decSetOCRUserTemplate (

OCREnable_t nEnable,

BYTE *pszTemplate

);

The decSetOCRTemplates function is used to define the template or combination of templates the decoder will use during an OCR decode attempt. Normally templates can be logically OR'd together in any combination using the bit values listed below to enable the desired templates. The one exception is the Passport template. When Passport mode is desired, only the Passport template may be enabled. Attempts to enable any other template along with the passport template will generate an error. Setting the nTemplates parameter to 0 has the same effect as disabling OCR decoding. Valid values for nTemplates can be from the table below:

Value	**Template**
0x01	User
0x02	Passport
0x04	ISBN
0x08	Price Field
0x10	MICR E-13B

Passport Template

The Passport Template may be used to read passports, visas and official travel documents based on the ICAO standard. This template reads both OCR-A and OCR-B fonts. Passports and Format-A visas each consist of two rows of 44 OCR-B characters.

Format-B visas and TD-2 travel documents each have two rows of 36 OCR-B characters, while TD-1 travel documents employ three rows of 30 OCR-B characters.

Example: Passport OCR-B text

```
P<UTOERIKSSON<<ANNA<MARIA<<<<<<<<<<<<<<<<<<<
L898902C<3UTO6908061F9406236ZE184226B<<<<<14
```

Example: Format-A Visa OCR-B text

```
V<UTOERIKSSON<<ANNA<MARIA<<<<<<<<<<<<<<<<<<<
L898902C<3UTO6908061F9406236ZE184226B<<<<<<<
```

Example: Format-B Visa OCR-B text

```
V<UTOERIKSSON<<ANNA<MARIA<<<<<<<<<<<
L898902C<3UTO6908061F9406236ZE184226
```

Example: TD-1 travel document OCR-B text

```
I<UTOD231458907<<<<<<<<<<<<<<<
3407127M9507122UTO<<<<<<<<<<<2
STEVENSON<<PETER<JOHN<<<<<<<<<
```

Example: TD-2 travel document OCR-B text

```
I<UTOSTEVENSON<<PETER<<<<<<<<<<<<<<<
D231458907UTO3407127M9507122<<<<<<<2
```

ISBN Template

The ISBN Template is used to read an International Standard Book Number (ISBN) in either OCR-A or OCR-B font.

Example: 13 Character ISBN format in OCR-A text

```
ISBN 0-8436-1072-7
```

This format consists of the 4 letter ISBN followed by 13 characters which include hyphens. The last digit is a Mod 11 checksum of 10 numbers (0-9), or an "X." All ISBN results are checked for a valid checksum.

Example: 17 Character ISBN format in OCR-A text

```
ISBN 978-0-571-08989-5
```

This format differs from the 13 character format in that the checksum is a Mod 10 checksum of 10 numbers (0-9) only.

You can enable multiple Pre-Defined OCR templates along with the ISBN template. See Custom OCR Templates on page 5.

Price Field Template

The Price Field is used in a number of applications including book pricing. The Price Field Template reads both OCR-A and OCR-B fonts. The format is as follows:

C1234 P5678E

The field begins with a 'C' and ends with an 'E.' The first part of the Price Field is a 'C' followed by four numeric digits. The second half begins with a currency character. The above example shows the letter 'P' but the Price Field template allows the following additional characters:

$€£¥

Following the currency character, a numeric grouping of 3, 4, 5, or 6 digits is followed by a terminating letter 'E.' The template reads both OCR-A and OCR-B fonts. The following examples can also be read when the Price Field Template is enabled:

C6712 $801E

C0217 €4399E

C0823 ¥31559E

C0331 £706213E

You can enable multiple Pre-Defined OCR templates along with the Price Field template. See Custom OCR Templates on page 5.

MICR E-13B Template

MICR E-13B consists of 14 characters: the numbers 0-9 and 4 control characters. The 4 control characters are known as TOAD (Transit, On Us, Amount and Dash), and are output in the following manner:

MICR Char.	Function	ASCII Char.	Decimal	Hex
⑆	Transit	A	65	0x41
⑇	Amount	B	66	0x42
⑈	On Us	C	67	0x43
⑉	Dash	D	68	0x44

MICR E-13B is used in financial applications, such as checks, to encode bank account numbers, bank routing numbers, check numbers, and other information on a single row. There are standard guidelines that address how data must be represented on checks and other financial documents, but there is a great deal of flexibility left to the discretion of the document designer.

The MICR E-13B Template reads any MICR string whose length is between 4 and 40 characters. Only one consecutive space is allowed in a template,. Since there are many checks produced where the MICR line contains fields separated by more than one space, these fields will be read and output as individual MICR strings. There is a broad range of strings that produce MICR output, so you should check for partial reads of MICR text where only part of the targeted MICR string is actually in the image presented to the scan engine.

The following examples can be read when the MICR E-13B Template is enabled:

⑆123456789⑆

⑈01235⑈ ⑆123456789⑆ 193412454⑈

⑈98765⑈ ⑆568123977⑆   67891788⑈70

Note that in the third example, there will be 2 separate output results because of the 4 space gap between the first and second fields.

You can enable multiple Pre-Defined OCR templates along with the MICR E-13B template. See Custom OCR Templates on page 5.

One of the standard fields within MICR E13-B is the routing field. It begins with the Transit symbol (A) and is followed by 9 numeric digits and a terminating Transit symbol. In some checks, the routing field is separated on each end by at least one space and can be read as a standalone field. This would be done by creaing the following template (see Custom OCR Templates on page 5):

1 4 x 4 1 5 1 4 9 x 4 1 0

If the routing field is part of a longer field (i.e., there is no space between either the leading or trailing transit character and other MICR data), then a custom template must be created to read those documents.

Custom OCR Templates

Result_t decSetOCRUserTemplate(

OCREnable_t nEnable,

BYTE* pszTemplate

);

The decSetOCRUserTemplate function is used to both enable\set OCR mode and to set the user defined template. The nEnable parameter can be one of the values from OCREnable_t struction. The pszTemplate is the pointer to the user template defined as per the template specifications detailed earlier in this document.

You can create a custom template, or character string that defines the length and content of OCR strings that will be read with your scan engine. The templates define the OCR font as well as the layout of the text in a row and column format. Each row can have up to 50 characters, with up to 18 rows in a template, with a maximum of 320 characters. Within each character position, the allowable characters can be specified either through explicit ASCII values, groups of ASCII values, wildcard characters, or combinations of these types. To achieve better OCR results, limit each character position's values to the specific expected values in your application.

Spaces

Internal gaps longer than one space are not allowed in a template. For example, the OCR text

```
ONE SPACE
```

is valid because there is only one space between the E and S in the text. However, the following text is illegal given the two spaces between the O and S:

```
TWO  SPACES
```

An arbitrary number of spaces at the beginning and end of a line are acceptable. These spaces must be included in the template with the ASCII value of a space (32 decimal, 0x20 hex), and not be included as part of a group or wildcard character.

Character Size

The ideal height of an OCR character after sampling is about 20 pixels, but characters up to 50 pixels in height can be read. If OCR characters are consistently above 40 pixels in height, downsampling the image by a factor of 2 will achieve better results in both speed and decode rates.

Euro, Pound, and Yen Currency Characters

7 bit ASCII values are used in the OCR template strings. However, there are no 7 bit ASCII representations for the euro, pound, or yen currency characters. 8 bit codes for these characters are:

Currency	**Decimal**	**Hex**
Euro	128	0x80
Pound	163	0xA3
Yen	165	0xA5

The hex character is output. For example, the euro output is [0xA3]. Refer to the ASCII Conversion Chart (page 17).

Creating a Custom OCR Template

Custom OCR Templates are strings made up of various control codes, along with standard ASCII values.

Control Codes Chart

Control Code	Value	Argument
End of Template	0	
New Template	1	Font: 1 - OCR-A 2 - OCR-B 3 - Both A & B 4 - MICR 5 - Semi
New Line	2	
Define Group Start	3	ID [001-255]
Define Group End	4	
Wildcard: Numeric	5	[0-9]
Wildcard: Alpha	6	[A-Z uppercase]
Wildcard: Alphanumeric	7	[0-9] [A-Z uppercase]
Wildcard: Any (including space)	8	
Defined Group	A	ID [001-255]
In Line Group Start	B	
In Line Group End	C	
Checksum	D	Weights, Type, MOD
Fixed Character Repeat	E	[01-50]
Variable Character Repeat	F	Range Low [01-50] Range High [01-50]
ASCII Hex Value	x##	2 digits

Note: In all following examples, spaces are used in template strings for readability only.

New Template

All OCR templates begin with the **New Template** control code. The value immediately following this control code indicates the font(s) for which this template is designed.

Example: You need to read 8 numeric digits in either OCR-A or OCR-B:

12345678

The string would be: 1 3 5 5 5 5 5 5 5 5 0

The breakdown:

Control Code	Description
1	New Template Code
3	Both OCR-A and OCR-B font
5	Wildcard: Numeric - 8 times
5	
5	
5	
5	
5	
5	
5	
0	End of Template

A template may contain multiple distinct templates all within the same string. Begin each template with a **New Template** control code.

Multiple Lines

A new line within a multple line template is indicated by the **New Line** control code.

Example: You need to read 2 lines of OCR-A characters. The first line has 4 numeric digits and the second line has 8 alphanumeric characters and spaces:

4321
A-3D FG9

The string would be: 1 1 5 5 5 5 2 8 8 8 8 8 8 8 8 0

The breakdown:

Control Code	Description
1	New Template Code
1	OCR-A font
5	Wildcard: Numeric - 4 times
5	
5	
5	
2	New Line
8	Wildcard: Any (including space) - 8 times
8	
8	
8	
8	
8	
8	
8	
0	End of Template

Repeating Characters

To simplify the creation of user templates, the **Fixed Character Repeat** control code may be used to repeat a character a specified number of times. Any specific ASCII value, wildcard, or group can be repeated. Because each OCR line is limited to a maximum of 50 characters, you can shorten your string by using a fixed character repeat.

Example: Using the example for New Template, page 7, you need to read 8 numeric digits in either OCR-A or OCR-B:

12345678

The string without repeating characters was: 1 3 5 5 5 5 5 5 5 5 0

Using Repeating Characters, it would be: 1 3 5 E 0 8 0

The breakdown:

Control Code	Description
1	New Template Code
3	Both OCR-A and OCR-B font
5	Wildcard: Numeric
E	Fixed Character Repeat - 8 times
0	
8	
0	End of Template

Variable Character Repeat

The **Variable Character Repeat** control code may be used to repeat a count for a character a variable number of times. Any specific ASCII value, wildcard, or group can be repeated.

The control code requires 4 bytes that give the minimum and maximum number of times (2 bytes each) that the character may appear in the template. Because each OCR line is limited to a maximum of 50 characters, you can shorten your string by using a variable character repeat. The minimum and maximum counts must be in the range from 1 to 50, with the minimum count less than or equal to the maximum count.

Example: You need to read OCR-B characters that may contain 5, 6, or 7 numeric digits. The string, without repeating variable characters, would be:

1 2 5 5 5 5 5 1 2 5 5 5 5 5 5 1 2 5 5 5 5 5 5 5 0

Using repeating variable characters, the template would be:

1 2 5 F 0 5 0 7 0

The breakdown:

Control Code	Description
1	New Template Code
2	OCR-B font
5	Wildcard: Numeric
F	Variable Character Repeat - 5 min, 7 max
05	
07	
0	End of Template

Groups

In a given character position, you must specify which values a text character may take. To reduce the overall size of templates, you may define common groups of ASCII characters and then use the defined group control character rather than repeating the same sequence over and over.

Groups can be made up of individual ASCII values or wildcard values. The wildcard values are Control Codes Numeric (5), Alpha (6), Alphanumeric (7), and Any(8).

To define a group, specify the **Defined Group** control code followed by an ID from 1 to 255. (Up to 255 groups may be defined in a single template.) Use the group ID to use the group in any template you build.

Note: Groups may not be nested.

Example: You need to read a 3 numeric digits, then either A, B, C, or another numeric digit. The string would be:

1 2 3 0 0 1 x 4 1 x 4 2 x 4 3 5 4 5 5 5 A 0 0 1 0

Note: Spaces are used in this example only for ease of readability.

The breakdown (dark box indicates group definition):

Control Code	Description
1	New Template Code
2	Both OCR-A and OCR-B font
3	Defined Group
001	Group ID
x41	ASCII hex value for A
x42	ASCII hex value for B
x43	ASCII hex value for C
5	Numeric Digit
4	Define Group End
5	3 Numeric Digits
5	
5	
A001	Defined Group, ID 001
0	End of Template

See the ASCII Conversion Chart on page 17 for character to hex conversions.

In Line Group

The **In Line Group** defines a one time instance of a group that occupies one character position in the template. Use this for unique groups of characters that occur only once.

Checksums and Weighting

A checksum reduces the probability of misreads. There are two types of checksums: row and block. For additional checksum protection, there are four different weighting schemes: 1, 12, 13, and 137. The checksum calculation is based on modulo arithmetic. The modulo factor may vary from 6 to 36.

The byte immediately following the **Checksum** control code (D) defines the type of checksum that will be used:

Checksum Table

Bit Position(s)	**Meaning**
7,6: Weight Scheme	00: Weight Scheme: 1
	01: Weight Scheme: 12
	10: Weight Scheme: 13
	11: Weight Scheme: 137
5: Checksum Type	0: Row
	1: Block
4-0: Modulo Value	Checksum Modulo - 5

Row Checksums (0) perform a checksum calculation on all characters preceding them up to the first character on the same row. Block Checksums (1) perform a checksum calculation on all characters up to the very first character in the template; they span multiple rows. The 5 bit Modulo Value stores the Checksum Modulo - 5. The stored number can range from 1, which is a Checksum Modulo value of 6, to 31, which describes a Checksum Modulo of 36. A Modulo value of 0 (Checksum Modulo of 5) is illegal. The characters within a checksum field have a numerical value that is used in the checksum calculation. Digits are converted to their numerical value (0-9), while uppercase letters range from 10 for an “A” to 36 for a “Z.” All punctuation characters have a value of 0 for checksum purposes. However, they do count as a spot for determining the weight values used in calculating the checksum.

Weight Scheme

The Weight Scheme defines how the values described above can be changed based on their character position. The default weight scheme is 1. This means that the checksum is based only on the character value and is not dependent on its position. The other weight schemes multiply the character value by a repetitive weight value that helps in identifying characters that have had their column locations switched. The 4 weight schemes are:

Weight Scheme Table

Weight Scheme	**Multiplier Values**
1	1 1 1 1 1 ...
12	1 2 1 2 1 2 ...
13	1 3 1 3 1 3 ...
137	1 3 7 1 3 7 1 3 7 ...

The checksum character always starts with a weight of 1. As you move to the left of the checksum, the weight value is updated to the next member of the sequence. The sequences repeat until the first character in a row for a Row type checksum, and to the first character in the template for a Block type checksum. The resulting sum is then divided by the Checksum Modulo number of the checksum. The remainder of this division should be zero for a valid checksum.

Checksum Examples

ABCD6

EFG5Y

The two lines of OCR-B text above both contain a row checksum. In addition, the last character of row 2 is a block checksum. The 2 row checksums are mod 10 with a 13 weight (133 decimal, 0x85 hex), while the block checksum is a mod 36 with a 137 weight (255 decimal, 0xFF hex). The following template will read this text:

1 2 6 6 6 6 **D 8 5** 2 6 6 6 **D 8 5 D F F** 0

Note: Bold text shows the row and block checksum notations.

The breakdown of the row checksum:

D85	Description
1	Weight Scheme: 13 (see Checksum Table, page 13)
0	
0	Checksum Type: Row (see Checksum Table, page 13)
0	Translation of the sum to binary code
0	
1	
0	
0	

The breakdown of the block checksum:

DFF	Description
1	Weight Scheme: 137 (see Checksum Table, page 13)
1	
1	Checksum Type: Block (see Checksum Table, page 13)
1	Translation of the sum to binary code
1	
1	
1	
1	

The top line checksum is the 6 at the end of the line. While this example shows the checksum at the end of the line, it may appear anywhere on the line and then protects all the characters to its left. The following sum is generated to verify a proper checksum on line 1:

6 D C B A

(1x6) + (3x13) + (1x12) + (3x11) + (1x10) = 100

Note that the 13 weight scheme starts with a 1 on the checksum digit, and then alternates between a 1 and 3 for all digits to the left of the checksum, up to the first character on the line. The numerical values of the alphabetic characters range from 10 for an 'A' to a 35 for a 'Z.' The sum of 100 is a multiple of 10, so the mod 10 checksum here has passed. On line 2, the row checksum is the 5 following the G. Verify its line by generating its sum:

```
5   G   F   E
```

(1x5) + (3x16) + (1x15) + (3x14) = 110

Again, a value is obtained that is a multiple of 10, validating this row checksum. The X at the end of the line is a mod 36 block checksum with 137 weighting. It protects all the characters in the template, including the first line. Calculating its sum working backwards from the block checksum and using the 137 weighting scheme:

```
Y   5   G   F   E   6   D   C   B   A
```

(1x34) + (3x5) + (7x16) + (1x15) + (3x14) + (7x6) + (1x13) + (3x12) + (7x11) + (1x10) = 396

The resulting sum is a multiple of 36, so the block checksum has been validated.

ASCII Conversion Chart

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	NUL	32	20		64	40	@	96	60	‘
1	01	SOH	33	21	!	65	41	A	97	61	a
2	02	STX	34	22	“	66	42	B	98	62	b
3	03	ETX	35	23	#	67	43	C	99	63	c
4	04	EOT	36	24	$	68	44	D	100	64	d
5	05	ENQ	37	25	%	69	45	E	101	65	e
6	06	ACK	38	26	&	70	46	F	102	66	f
7	07	BEL	39	27	‘	71	47	G	103	67	g
8	08	BS	40	28	(	72	48	H	104	68	h
9	09	HT	41	29	)	73	49	I	105	69	i
10	0A	LF	42	2A	*	74	4A	J	106	6A	j
11	0B	VT	43	2B	+	75	4B	K	107	6B	k
12	0C	FF	44	2C	,	76	4C	L	108	6C	l
13	0D	CR	45	2D	-	77	4D	M	109	6D	m
14	0E	SO	46	2E	.	78	4E	N	110	6E	n
15	0F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	DC1	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	DC3	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[	123	7B	{
28	1C	FS	60	3C	<	92	5C	\	124	7C	|
29	1D	GS	61	3D	=	93	5D	]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	80	€	160	A0		192	C0	À	224	E0	à
129	81	□	161	A1	¡	193	C1	Á	225	E1	á
130	82	‚	162	A2	¢	194	C2	Â	226	E2	â
131	83	ƒ	163	A3	£	195	C3	Ã	227	E3	ã
132	84	„	164	A4	¤	196	C4	Ä	228	E4	ä
133	85	…	165	A5	¥	197	C5	Å	229	E5	å
134	86	†	166	A6	¦	198	C6	Æ	230	E6	æ
135	87	‡	167	A7	§	199	C7	Ç	231	E7	ç
136	88	ˆ	168	A8	¨	200	C8	È	232	E8	è
137	89	‰	169	A9	©	201	C9	É	233	E9	é
138	8A	Š	170	AA	ª	202	CA	Ê	234	EA	ê
139	8B	‹	171	AB	«	203	CB	Ë	235	EB	ë
140	8C	Œ	172	AC	¬	204	CC	Ì	236	EC	ì
141	8D	□	173	AD	-	205	CD	Í	237	ED	í
142	8E	Ž	174	AE	®	206	CE	Î	238	EE	î
143	8F	□	175	AF	¯	207	CF	Ï	239	EF	ï
144	90	□	176	B0	°	208	D0	Ð	240	F0	ð
145	91	‘	177	B1	±	209	D1	Ñ	241	F1	ñ
146	92	’	178	B2	²	210	D2	Ò	242	F2	ò
147	93	“	179	B3	³	211	D3	Ó	243	F3	ó
148	94	”	180	B4	´	212	D4	Ô	244	F4	ô
149	95	•	181	B5	µ	213	D5	Õ	245	F5	õ
150	96	–	182	B6	¶	214	D6	Ö	246	F6	ö
151	97	—	183	B7	·	215	D7	×	247	F7	÷
152	98	˜	184	B8	¸	216	D8	Ø	248	F8	ø
153	99	™	185	B9	¹	217	D9	Ù	249	F9	ù
154	9A	š	186	BA	º	218	DA	Ú	250	FA	ú
155	9B	›	187	BB	»	219	DB	Û	251	FB	û
156	9C	œ	188	BC	¼	220	DC	Ü	252	FC	ü
157	9D	□	189	BD	½	221	DD	Ý	253	FD	ý
158	9E	ž	190	BE	¾	222	DE	Þ	254	FE	þ
159	9F	Ÿ	191	BF	¿	223	DF	ß	255	FF	ÿ

Technical Assistance

If you need assistance installing or troubleshooting your device, please contact us by using one of the methods below:

Knowledge Base: www.hsmknowledgebase.com

Our Knowledge Base provides thousands of immediate solutions. If the Knowledge Base cannot help, our Technical Support Portal (see below) provides an easy way to report your problem or ask your question.

Technical Support Portal: www.hsmsupportportal.com

The Technical Support Portal not only allows you to report your problem, but it also provides immediate solutions to your technical issues by searching our Knowledge Base. With the Portal, you can submit and track your questions online and send and receive attachments.

Web form: www.hsmcontactsupport.com

You can contact our technical support team directly by filling out our online support form. Enter your contact details and the description of the question/problem.

Telephone: www.honeywellaidc.com/locations

For our latest contact information, please check our website at the link above.

Honeywell Scanning & Mobility
9680 Old Bailes Road
Fort Mill, SC 29707

www.honeywellaidc.com

OEM-API-OCR-UG Rev A
10/12