ThorX7.One Integrated Industrial Marking System

User Manual for TX7 Instruction

Catalogue

Preface

Purpose

This manual is intended to provide guidance for user programming using TX7 directives

Basic knowledge required

-This manual is intended for programmers, operators, and maintenance and repair personnel.

-In order to understand this manual well, basic computer programming knowledge is required.

-In addition, knowledge of computers and computer operating systems or other work equipment similar to computers is required.

Scope of application of this manual

This manual is suitable for ThorX7.One integrated industrial marking system [external communication control module V1.0] and subsequent upgrades.

1 Brief introduction of TX7 instruction

TX7 instruction is a set of instructions composed of strings, which is related to case. TX7 instruction is designed to control the operation process of ThorX7.One integrated industrial markup system. TX7 instruction does not have the logic ability of computer programming language. Users can embed TX7 instructions in relevant programming languages according to their own requirements to achieve the purpose of automatic control.

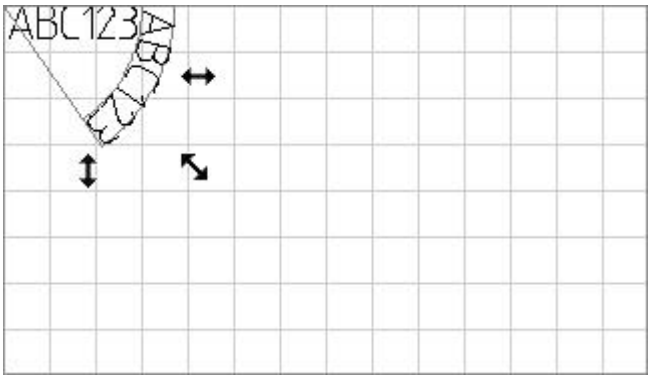
2 Quick start

1. The serial port of PC machine (or similar equipment) is connected with the external control serial port of ThorX7.One integrated industrial marking system (ThorX7 equipment) by using RS232 connection wire.
2. Power the ThorX7 device, wait for the ThorX7 system to start, click the start button on the screen, and enter the ThorX7 interface.
3. Run a serial port debugging software on PC (or other serial port software that can send and receive ASCII code), set communication rate: 9600, data bit: 8, parity check: no, stop bit: 1, stream Control: none, then open the serial port connected to the ThorX7 device.
4. Enter "list type" in the serial debugging software and send it in the form of ASCII code (non-HEX16 mode). If the connection is normal, you should be able to receive something like "TxMark7PiContent_Text,TxMark7PiContent_BowText,TxMark7PiContent_DataMatrix.". The "list type" instruction is used to check the connected Tho What tag types can rX7 devices support. Note that the TX7 instruction is case dependent, meaning

that each instruction including its feedback must comply with the letter case, such as the "List Type" instruction, which may receive feedback from "! Bad Command:List".

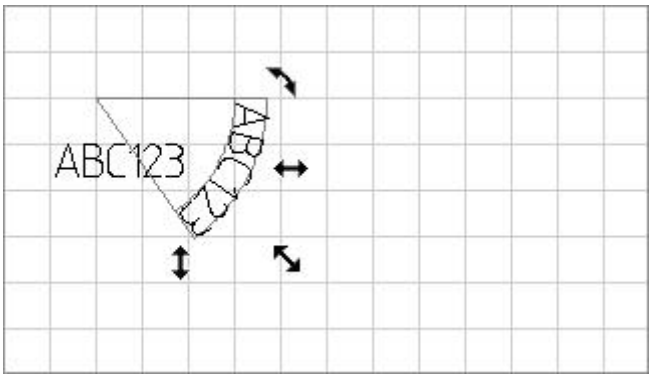
5. Now use the instruction to create a new text tag on the ThorX7 device. Send the instruction "new TxMark7PiContent_Text" and observe the ThorX7 device when a standard text appears in the upper-left corner of the workspace This type is marked and is selected for the default "ABC123". The "new" instruction here represents a new tag, the parameter "TxMark7PiContent_Text" represents a text type, and "new TxMark7PiContent_Text" means creating a new standard text type tag. After the instruction is executed successfully, the serial port debugging software of PC side will receive a feedback, the feedback content is "newmark1", "newmark1" is the name of the new tag for ThorX7 device.

6. According to this method, a fan text tag is established, and the instruction is "new TxMark7PiContent_BowText". After successful execution, a fan text tag will appear on the ThorX7 device, and the serial port debugging software on the PC side will receive feedback "newmark2". The name for this sector text tag is "newmark2", and the display on the ThorX7 device should look like this, as shown in the following figure:



7. To modify the parameter of a tag, you must know the name of the tag and select the tag. Now to change the location of the "newmark1" tag, first, select the tag using the selection instruction, send the instruction "select newmark1", the feedback "newmark1" represents that the tag "newmark1" has been selected successfully, if you select a tag that does not exist, For example, the instruction "select newmark3" will

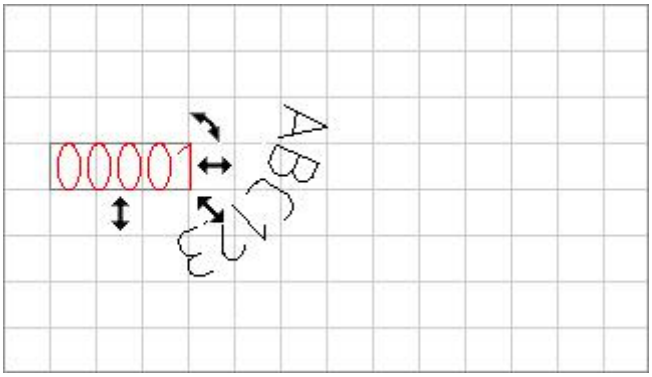
feedback "% null", indicating that none of the tags have been selected. Because the instruction selected the wrong marker Remember, now that both tags on the ThorX7 device are in a non-selective state, re-use "select newmark1" to select the "newmark1" text tag that was previously created. The instruction "set Pos=10,30" is then sent to modify the location of the tag, the "set" instruction is used to modify the various attributes of the tag, the "Pos" parameter represents the location attribute, and the "10N30" following "=" is the X coordinate and the Y coordinate, respectively. Feedback on the PC side "newmark1" represents the success of the "newmark1" tag New properties set. In this way, you can also modify the location of "newmark2" and send subsequent instructions in turn: "select newmark2" and "set Pos=30,20". The display on the ThorX7 device should look like this, as shown in the following figure:



8. Now change the font size of these two tags. If you want to modify the attributes of multiple tags at the same time, you need to select multiple tags at the same time, use "select newmark1,newmark2" to select the two previously created tags, the "select" directive supports the selection of multiple tags, and the names of each tag are ", "separate. Also using the "set" instruction to modify the font properties of the two tags, send the instruction "set CharHeight=10", you can see that the word height of both tags is set to 10mm, send instruction. Set CharWidth=7 ", you can see that the word width of both tags is set to 7mm, and that" CharHeight "and" CharWidth "correspond to the character height and width attributes of the tag, respectively.

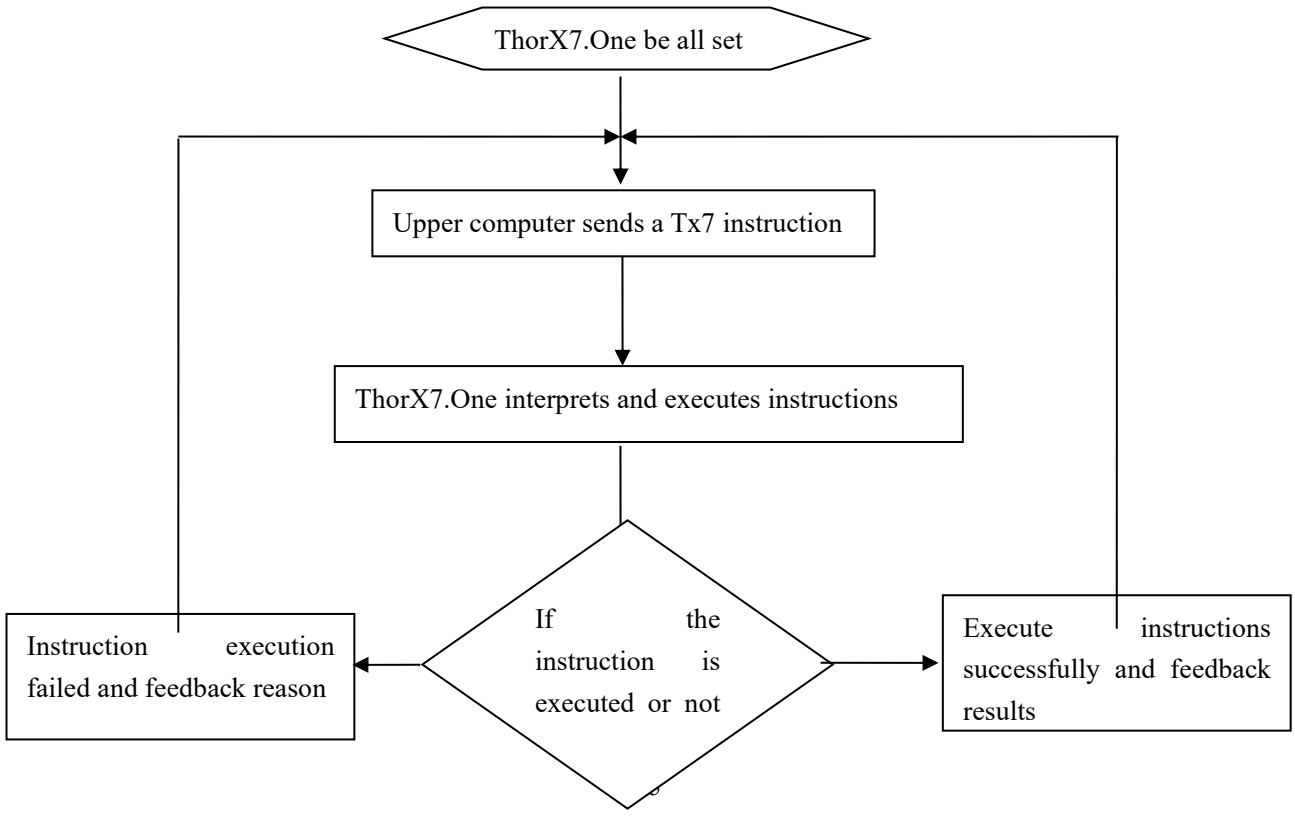
9. To modify the content of "newmark1", send the subsequent instructions "select newmark1" and "set Content=00001" in turn, and now the content of "newmark1" has been changed to "0001".

10. Finally, set "newmark1" to a pipeline number, select the "newmark1" tag, and use the instruction "code TxMark7PiCoder_SerialNumber" to set "newmark1" to a pipeline number. The "code" instruction is used to set the encoding of the tag. "TxMark7PiCoder_SerialNumber" represents the pipeline code. At this point, the display on the ThorX7 device should look like this, as shown in the following figure:



11. Here, assuming that everything we need to orchestrate is complete, send an instruction "start" to control the ThorX7 device to immediately start a print.

3 Instruction execution flow



4 Instruction introduction

Composition: *instruction parameters*

Note: *instructions and parameters need to be separated by a space*

Parameter convention: *surrounded by a < > symbol, represents a required option.*

Surrounded by the [] symbol, representing optional.

Normal feedback: *string*

Error feedback:! *Starting string or% null*

Common feedback explanations:

!Miss Parameter *Instruction lacks the necessary parameters*
!Invalid Parameter *Invalid Parameter*
!Bad Command *Bad Command*
%null *Empty*

4.1 list: *list instruction*

Format: list < parameters >

1. List all tags in the current document

Instruction: **list mark**

Feedback: Tag name 1,tag name 2,tag name n...

%null: *No tags in the document*

2. Lists selected tags in the current document

Instruction: **list selmark**

Feedback: Tag name 1,tag name 2,tag name n...

%null: *No tags in the document*

3. List supported tag types (see Appendix 1 for tag types)

Instruction: **list type**

Feedback: Tag type 1, tag type 2, tag type n...

%null: *No tag type available*

4. ist the types of coding supported (see Appendix 2 for coding types)

Instruction: list coder

Feedback: Encoding type 1, encoding type 2, encoding type n...

%null: *No coding type available*

4.2 new: *new tag instruction*

Format: new <Tag type>

Note: see Appendix 1 for tag types

Note: drawing markup is not supported

Feedback: **newmarkN:** *The name of the tag after the new creation is successful*

!Invalid Type: *Invalid tag type*

!Unsupported Type: *The tag type does not support the new instruction*

Eg: Assume that the current ThorX7 device supports text type tags

Instruction: new TxMark7PiContent_Text

Feedback: newmark1

4.3 del: *delete tag instruction*

Format: del [Tag name 1,tag name 2,tag name N...]

Note: When there is no parameter, the default is to delete the mark in the selected state.

Feedback: The set of tag names that were successfully deleted, separated by ","

%null: *No one tag was deleted*

Eg 1: Assume that there are two newmarks, newmark1 and newmark2 in the document.

Instruction: del newmark1,newmark2

Feedback: newmark1,newmark2

Eg 2: Assume that there are newmark2, newmark3 in the document, but there is no newmark1.

Instruction: del newmark1,newmark2,newmark3

Feedback: newmark2,newmark3

4.4 select: *select mark instruction*

Format: select <Tag name 1,tag name 2,tag name N...>

Feedback: The set of tag names that were selected successfully, separated by ","

%null: *No one tag is selected*

Eg 1: Assume that there are two newmarks, newmark1 and newmark2 in the document.

Instruction: sel newmark1,newmark2

Feedback: newmark1,newmark2

Eg 2: Assume that there are newmark2, newmark3 in the document, but there is no newmark1.

Instruction: sel newmark1,newmark2,newmark3

Feedback: newmark2,newmark3

4.5 code: *Set encoding instructions*

Format: code <Coding type>

Description: Set the encoding for the tag selected in the current document

Remarks: See Appendix 2 for coding types.

Feedback: The set of tag names that were successfully set to be encoded, separated by ","

!No Selected Mark: *No selected tags are available for processing*

!Invalid Coder: *Invalid encoding type*

%null: *No tag was successfully encoded*

Eg: Assume that the current ThorX7 device supports serial number encoding and that a text type tag has been selected in the document.

Instruction： code TxMark7PiCoder_SerialNumber

4.6 set: *Attribute setting instruction*

Format: set <Attribute name=attribute value>

Description: Set attributes for the tag selected in the current document

Remarks: See Chapter 4 and Chapter 5 for attribute names.

Feedback: The set of tag names that were successfully set by the attribute, separated by ","

!No Selected Mark: *No selected tags are available for processing*

!Invalid Properties: *Invalid attribute*

!Unsupport Properties: *Unsupported attribute*

%null: *No tag is set property*

Eg: Assume that there is a text type tag "newmark1" in the current document and is in the selected state.

Instruction: set CharHeight=10

Feedback: newmark1

4.7 get: *Get attribute instruction*

Format: get <Attribute name>

Description: Get the attribute value of the selected status tag in the current document

Note: The returned properties are sorted by the selected tag name.

Feedback: attribute value 1, attribute value 2, attribute value n...

!No Selected Mark: *No selected tags are available for processing*

!Unsupport Properties: *Unsupported attribute*

%null: *No tag returns a valid attribute value*

Eg 1: Assume that there are two text type tags "newmark1" and "newmark2" in the current document and they are all selected. The word height of "newmark1" is 10, and the word height of "newmark2" is 7.

Instruction: get CharHeight

Feedback: 10,7

Eg 2: Assume that there are two text type tags "newmark1" and "newmark3" in the current document, and a picture tag "newmark2", the word height of "newmark1" is 10, the word height of "newmark3" is 7, and the "newmark2" picture The tag does not support the word height attribute, and all three tags are selected.

Instruction: get CharHeight

Feedback: 10,,7

Eg 3: Assume that there is only one picture tag "newmark2" in the current document, and the "newmark2" picture tag does not support the word height attribute, this flag is selected.

Instruction: get CharHeight

Feedback: !Unsupport Properties:CharHeight

4.8 newfile: *New file instruction*

Format: newfile

Feedback: **done**: *New file completion*

!failed: *New file failed*

4.9 open: *Open file command instruction*

Format: open <file name>

Feedback: **file name**

!failed: *fail to open the file*

4.10 save: *Save file instruction*

Format: save <file name>

Feedback: **file name**

!failed: *Failed to save file*

4.11 start: *Engraving instructions*

Format: start

Feedback: **ok**: *Marking begins*

print completed: *Marking completed*

!print break: *Engraving is interrupted*

4.12 reset: *Reset instruction*

Format: reset

Feedback: **ok**: *Reset start*

completed: *Reset completed*

!reset failed: *Reset failed*

!reset break: *Reset is interrupted*

4 Tag attribute

1.Universal tag attribute

Name	*Tag name, string, must start with a letter.*
Enable	*Whether to imprint, Boolean, value 1 or 0.*
PassLine	*Whether it is an empty trace, Boolean, the value is 1 or 0.*
Index	*The engraved serial number of the mark, the integer value type.*
Pos	*The reference coordinates of the marker, in the format "x, y, z", floating point, in mm.*
Content	*Tag content, string.*

2.Text tag attribute

Font	*Font, numeric, representing the number of the font.*
FontEx	*Extended font, numeric, representing the number of the font.*

CharHeight *Word height, floating point type, unit mm.*

CharWidth *Word width, floating point type, unit mm.*

CharSpacing *Word spacing, floating point, in mm.*

Arrange *Arrange between words, 0=standard arrangement, 1=compact arrangement, 2=equal width arrangement*

3.Sector text tag attribute

The fan-shaped text tag has all the attributes of the text tag, and additionally has the following attributes:

Radius *Fan radius, floating point type, in mm.*

Entad *Character direction, 0=outward, 1=inward*

4.QR code tag attribute

DotSp *Point spacing, floating point, unit mm*

DotSize *Point size, floating point type, unit mm*

5 Coding attribute

1.serial number encoding attribute

Increment *Serial number increment, integer value type.*

Repeat *The number of repetitions, the integer value type.*

MinValue *Minimum value, the integer value type*

MaxValue *Maximum value, the integer value type*

Reset *Auto zero, 0 = disable, 1 = daily zero, 2 = monthly zero.*

TabooNum *Taboo number, integer value type.*

Hex *Binary, 0=10, 1=hex*

2.VIN, PIN encoding attribute

Increment *Serial number increment, integer value type.*

SerialLength	*Serial number length, integer value type.*
Repeat	*The number of repetitions, the integer value type.*
MinValue	*Minimum value, the integer value type*
MaxValue	*Maximum value, the integer value type*
TabooNum	*Taboo number, integer value type.*

Appendix 1 Marker Type

TxMark7PiContent_Text	*Text*
TxMark7PiContent_BowText	*Scalloped text*
TxMark7PiContent_DataMatrix	*QR code*
TxMark7PiContent_Graph	*Graphics*
TxMark7PiContent_Chart	*Symbol*
TxMark7PiContent_Ruler	*Ruler*
TxMark7PiContent_Beeline	*Straight line ("new" instruction is not supported)*
TxMark7PiContent_Bezier	*Bezier curve ("new" instruction is not supported)*
TxMark7PiContent_Rectangle	*Rectangle ("new" instruction is not supported)*
TxMark7PiContent_Circle	*Round ("new" instruction is not supported)*
TxMark7PiContent_3PArc	*Three-point arc ("new" instruction is not supported)*

Appendix 2 Encoding Type

TxMark7PiCoder_Empty	*No coding*
TxMark7PiCoder_SerialNumber	*Serial number*
TxMark7PiCoder_VIN	*VIN code*
TxMark7PiCoder_PIN	*PIN code*
TxMark7PiCoder_Date	*Date*
TxMark7PiCoder_XCode	*Composite coding*