1 协议基础

本章介绍了Pepperl + Fuchs扫描数据协议（PFSDP）的基础知识。

1.1 基本设计

通信协议规范基于以下基本设计决策：
提供了使用 HTTP 请求（和响应）的简单命令协议以便参数化和控制传感器。 可以使用标准 Web 浏览器或通过建立到 HTTP 端口的临时 TCP / IP 连接来访问 HTTP。
使用单独的 TCP / IP 或 UDP / IP 通道从传感器接收传感器过程数据（扫描数据）。对于需要安全和错误证明扫描数据传输的每个应用程序，建议使用 TCP 通道。对于需要扫描数据的最小延迟传输的应用，建议使用 UDP 通道。
扫描数据的输出始终符合以下约定：
数据输出被执行为具有适于公共以太网帧大小（TCP 以及 UDP）的分组大小的分组。
单个分组总是包含仅单次连续扫描的数据。扫描数据输出始终从每次（新）扫描的新数据包开始。
对于扫描数据输出，用户应用程序可以从具有不同级别的信息细节的多个数据类型中进行选择。
这样，客户端可以决定仅接收其单独应用所需的数据量-减少流量。此外，这提供了实现对扫描数据输出的未来扩展（例如，添加附加信息）的简单方法。
所有二进制数据的字节顺序是小端字节（最低有效字节优先）。传感器和 PC CPU 的 DSP 都使用 Little Endian - 因此不需要进行转换。
传感器不严格限制活动客户端连接的数量和客户端请求的（扫描）数据量。基本上用户负责设计他的（客户端）系统或应用程序，传感器可以处理请求的数据量，而不会过载。

1.2 HTTP 命令协议

HTTP 命令协议提供了一种简单的统一方法来控制应用软件的传感器操作。HTTP 命令用于配置传感器测量以及读取和更改传感器参数。此外，它可用于设置（并行）提供传感器扫描数据的 TCP 或 UDP 数据通道。
本节介绍基本的 HTTP 命令协议和用户可用的各种命令。使用附加的 TCP 或 UDP 通道传输扫描数据在 3.4 节中说明。
请注意：
R2000 提供对 HTTP / 1.1 的完全支持 - 但是目前不支持持久连接（根据 HTTP / 1.1 标准[4]是可选的）。每个 HTTP 响应包括“连接：关闭”报头，以通知客户端后续 HTTP 请求需要到传感器的单独的 TCP / IP 连接。

1.2.1 发送命令

使用 RFC 2616 [4]定义的超文本传输协议（HTTP）向传感器发送命令。每个 HTTP 命令根据 RFC 3986 [7]构造为统一资源标识符（URI），具有以下基本结构：

< scheme >:< authority >/< path >?< query > # < fragment >

传感器的典型 HTTP 请求如下所示：

http://< sensor IP address >/cmd/< cmd_name >?< argument1 = value > & < argument2 = value >

因此，就 URI 而言，有效的 HTTP 命令由以下部分组成：

scheme总是'http://'

authority由传感器的 IP 地址（和端口号，如果需要）表示，

path由前缀′cmd /′和所请求的命令的名称（′< cmd_name >′）组成

query列出特定命令的其他参数

fragment当前未使用 - 哈希标记之后的任何内容都将被忽略

1.2.2 查询参数编码

命令 URI 的查询部分（参见第 1.2.1 节）用于将附加参数传输到 HTTP 命令（符合 RFC 3986 [7]）。本节描述了参数的组成为“key = value”对。

HTTP 命令参数使用以下方案（“key = value”对）组成：

key = value[; value][&key = value]

key表示接收一个或多个值的参数。 单个参数的多个值由分号";"分隔。单个命令需要多个参数，用＆符号分隔。

1.2.3 对命令的回复

向传感器发送命令后，可以接收以下回复：

HTTP 状态代码

HTTP 命令将首先使用标准 HTTP 状态代码进行应答。该代码指示命令（即 URI）是否已知并且已被正确接收。仅当 URI 无效或无法处理时，才会返回错误代码。有关 R2000 使用的 HTTP 状态代码的详细说明，请参见第 1.2.5 节。

命令错误代码

通常，HTTP 状态代码读为“OK”。在这种情况下，可以使用两个返回值来评估命令处理的结果：error_code和error_text。error_code包含用于命令调用的数字结果代码，而error_text提供文本错误描述。 这两个值都使用 JSON 编码[9]返回。第 1.2.6 节提供了所有 R2000 命令错误代码的详细说明。

命令回复数据

命令回复的主体包含任何请求的有效载荷数据。 此数据始终使用 JSON 编码[9]传输。 可能使用 base64 编码的 JSON 数组输出大量数据。

1.2.4 HTTP 请求和回复 - 低级示例

此部分显示了一个示例，说明如何在不使用 Web 浏览器的情况下将 HTTP 请求传输到传感器。假设将发送以下 HTTP 请求：

http://< sensor IP address >/cmd/get_parameter?list = scan_frequency

此请求被转换为一个简单的字符串（在本例中使用 HTTP / 1.0）：

1 GET /cmd/get_parameter?list = scan_frequency HTTP/1.0\r\n\r\n

然后将该字符串作为TCP / IP分组的有效载荷数据发送到传感器。传感器然后发回具有HTTP答复的 TCP / IP 分组作为有效载荷数据。HTTP 回复可以解析为具有以下内容的简单文本字符串：

1 HTTP/1.0 200 OK\r\n
2 Expires: -1\r\n
3 Pragma: no-cache\r\n
4 Content-Type: text/plain\r\n
5 Connection: close\r\n
6 \r\n
7 {\r\n
8 "scan_frequency":50,\r\n
9 "error_code":0,\r\n
10"error_text": "success"\r\n
11 }\r\n

此 HTTP 回复的最重要的部分是第一行包含 HTTP 错误代码，最后几行包含请求的信息包裹在一个 JSON 编码[9]对象中，由一对大括号表示。

1.2.5 HTTP 状态码

下表列出了设备使用的所有 HTTP 状态代码：

status code	message	description
200	OK	request successfully received
400	Bad Request	wrong URI syntax
403	Forbidden	permission denied for this URI
404	Not Found	unknown command code or unknown URI
405	Method not allowed	invalid method requested (currently only GET is allowed)

导致 HTTP 错误的（无效）查询示例

request	status code	error message
http://<ip>/cmd/nonsense	400	"unrecognized command"
http://<ip>/cmd/get_parameter&test	400	"unrecognized command"
http://<ip>/cmd/get_parameter?list	400	"parameter without value"
http://<ip>/test	404	"file not found"
http://<ip>/test/	404	"file not found"
http://<ip>/test/file	404	"file not found"

1.2.6 传感器错误代码

下表列出了设备返回的一些常规错误代码(error_code)：

error code	description (error_text)
0	"success"
100	"unknown argument '%s'"
110	"unknown parameter '%s'"
120	"invalid handle or no handle provided"
130	"required argument '%s' missing"
200	"invalid value '%s' for argument '%s'"
210	"value '%s' for parameter '%s' out of range"
220	"write-access to read-only parameter '%s'"
230	"insufficient memory"
240	"resource already/still in use"
333	"(internal) error while processing command '%s'"

（无效）命令的示例激发传感器错误代码

command (query)	code	error message
/cmd/get_protocol_info?list=test	100	"Unknown argument 'list'"
/cmd/get_parameter?list=test	110	"Unknown parameter 'test'"
/cmd/start_scanoutput	120	"Invalid handle or no handle provided"
/cmd/start_scanoutput?handle=test	120	"Invalid handle or no handle provided"
/cmd/set_parameter?ip_address=777	200	"Invalid value '777' for argument 'ip_address'."
/cmd/set_parameter?scan_frequency=999	210	"Value '999' for parameter 'scan_frequency' is out of range."
/cmd/set_parameter?serial=123456	220	"Write-access to read-only parameter 'serial'"

1.2.7 协议信息（get_protocol_info）

以太网协议用户应该意识到，根据协议版本，一些命令可能不可用或可能显示不同的行为。因此，用户应用程序应始终使用专用命令get_protocol_info检查协议版本，该命令返回通信协议的基本版本信息：

parameter name	type	description
protocol_name	string	Protocol name (currently always 'pfsdp')
version_major	uint	Protocol major version (e.g. 1 for 'v1.02', 3 for 'v3.10')
version_minor	uint	Protocol minor version (e.g. 2 for 'v1.02', 10 for 'v3.10')
command_list	string	List of all available HTTP commands

本文档涉及协议版本“1.01”，由 R2000 固件版本 1.20 和更新版本实现。

例

Query: http://< sensor IP address >/cmd/get_protocol_info

Reply: {


```
"protocol_name":"pfsdp",
"version_major":1,
"version_minor":1,
"commands":[
"get_protocol_info",
"list_parameters",
"get_parameter",
"set_parameter",
"reboot_device",
"reset_parameter",
"request_handle_udp",
"request_handle_tcp",
"feed_watchdog",
"get_scanoutput_config",
"set_scanoutput_config",
"start_scanoutput",
"stop_scanoutput",
"release_handle"
],
"error_code":0,
"error_text":"success"
}
```

2 使用 HTTP 的传感器参数化

2.1 参数类型

传感器提供对不同类型参数的访问。下表提供了相关类型的快速概述，更详细的描述在以下单独的子部分中：

type	description
`enum`	enumeration type with a set of named values (strings)
`bool`	boolean values (`on` / `off`)
`bitfield`	a set of boolean flags
`int`	signed integer values
`uint`	unsigned integer values
`double`	floating point values with double precision
`string`	strings composed of UTF-8 characters
`ipv4`	Internet Protocol version 4 addresses or network masks
`ntp64`	NTP timestamp values
`binary`	Binary data

与其类型无关，每个参数属于以下访问组之一：

access	description
sRO	static Read-Only access (value never changes)
RO	Read-Only access (value might change during operation)
RW	Read-Write access (non-volatile storage)
vRW	volatile Read-Write access (lost on reset)

大多数传感器参数存储在非易失性存储器中。因此，即使器件遇到了电源循环，它们的值仍然存在。请注意，非易失性存储器仅具有有限数量的写周期（典型> 10,000 个周期）。因此，只有在必要时才应写入所有非易失性参数。

2.1.1 枚举值(枚举)

使用枚举值的参数注意事项（枚举）：
枚举类型参数接受预定义值列表中的单个值。
每个枚举值由一个字符串（'named'值）定义。
每个枚举值通常（但不一定）对于特定参数是唯一的。
每个枚举参数一次只能保存一个值。
URI：指定的枚举值仅使用非保留的 ASCII 字符，并且当指定为 URI 上的命令的参数时，不需要百分比编码[7]。

2.1.2 布尔值（bool）

使用布尔值的参数注释（bool）：
布尔参数是枚举参数的特殊情况。
只接受指定的值 on 和 off。
每个bool参数一次只能保存一个值。

2.1.3 位字段（位字段）

使用位字段（位字段）的参数说明：
位字段将多个布尔标志组合成无符号整数值。
每个标志占据整数的单个位。
不是整数的每个位都需要分配给一个标志。
位可能被标记为保留。这些应该始终为零。
使用整数表示读取和写入位字段参数。

2.1.4 整数值（int，uint）

使用有符号整数值（int）和无符号整数值（uint）的参数的注释：
除非另有说明，整数值的值范围限于 32 位。
写入值时接受前导零（它们将被忽略）。
不支持十六进制或整数值的八进制表示。

2.1.5 双精度浮点值（双精度）

使用双精度浮点值的参数注意事项（双精度）：
点"."用作十进制标记（将小数部分与双精度数的整数部分分隔开）。
访问双精度浮点值时应使用浮点十进制格式（xxx.yyy）。不支持浮点指数格式（xxx.yyy Ezzz）。
双精度浮点值的小数部分的有效数字的数量可能对于一些参数是有限的。超出的数字被舍入或舍弃。

2.1.6 字符串值（字符串）

使用字符串值的参数注意事项（字符串）：
字符串表示设置的字符。
字符串的所有字符都需要以 UTF-8 格式编码[6]。
字符串的最大大小通常有限。有关实际尺寸限制，请参阅具体参数的说明。
URI：对于对字符串参数的写访问，其新值由 URI 中的周围的'='和'＆'隐式分隔（参见 RFC 3986 [7]）。任何额外添加的定界符（例如'" '）将被解释为字符串的一部分。
URI：在 URI 中保留一些字符，需要进行百分比编码[7]（详见第 1.2.2 节）。
当解析 UTF-8 编码的命令 URI 中的字符串类型的参数时，传感器执行以下步骤：

1.将 URI 解析成各个部分
2.解析编码字符的百分比
3.检查字符串的有效 UTF-8 编码
4.处理字符串（UTF-8 字节），例如。将其存储在非易失性存储器中
当传感器以 JSON 格式输出字符串类型的参数时，它对以下保留的 UTF-8 字符应用转义（如 RFC 7159 [9]第 2.5 节所要求）：

character	code	replacement
backspace	U+0008	'\b'
tabulator	U+0009	'\t'
new line	U+000A	'\n'
form feed	U+000C	'\f'
carriage return	U+000D	'\r'
double quote	U+0022	'\"'
solidus	U+002F	currently not replaced
backslash	U+005C	'\\'
other control characters	U+0000 ... U+001F	'\uXXXX'

2.1.7 IPv4 地址和网络掩码值（IPv4）

有关使用 IPv4 网络地址和子网掩码（IPv4）的参数的注意事项：
地址和网络掩码需要遵循互联网协议规范（RFC-791 [1]）的规则
地址被表示为字符串值，以人类可读的点分十进制表示法（即 10.0.10.9）
子网掩码表示为字符串值，以人类可读的点分十进制表示法（即 255.255.0.0）

2.1.8 NTP 时间戳值（ntp64）

使用 NTP 时间戳记的参数注意事项（ntp64）：
NTP 时间戳是 RFC 1305 [2]定义的网络时间协议的一部分。
NTP 时间戳表示为参考特定时间点的 64 位无符号定点整数（uint64）（以秒为单位）。最高有效 32 位表示整数部分（秒），低 32 位表示小数部分。
绝对时间戳（同步时间）是指自 1900 年 1 月 1 日以来所经过的时间。
相对时间戳（原始系统时间）是指从传感器通电以来所经过的时间。

2.1.9 二进制数据（二进制）

使用二进制数据的参数注意事项（二进制）：
二进制类型的参数存储二进制数据，而没有其他数据特定知识。参数值被简单地处理为字节的集合。二进制数据的解释是针对单个参数的。有关详细信息，请参阅具体参数的说明。
对于二进制参数，通常只执行大小检查。数据的最大大小取决于具体的参数。
对二进制参数的读取访问在 JSON 回复中返回其作为 base64 编码字符串的值（参见第 1.2.3

节）。
base64 编码将 8 位数据流转换为一个字符串，该字符串具有 64 个 ASCII 字符（6 位）的特定集合，这些字符对于大多数字符编码是可打印和通用的（详见 RFC 4648 [8]）。此编码需要 33％的存储空间。
对二进制参数的写访问需要将二进制数据编码为 URI 上的 base64url [8]字符串。 base64url 编码非常类似于 base64 编码，但使用略有不同的字符集，避免使用保留的 URI 字符。

2.2 传感器参数化的命令

本节介绍可用于操纵全局传感器参数的所有命令。

2.2.1 list_parameters - 列表参数

命令list_parameters返回所有可用的全局传感器参数的列表。
例
Query: http://< sensor IP address >/cmd/list_parameters

```
Reply: {
       "parameters":[
       "vendor",
       "product",
       "part",
       "serial",
       "revision_fw",
       "revision_hw",
       "max_connections",
       "feature_flags",
       "radial_range_min",
       "radial_range_max",
       "radial_resolution",
       "angular_fov",
       "angular_resolution",
       "ip_mode",
       "ip_address",
       "subnet_mask",
       "gateway",
       "scan_frequency",
       "scan_direction",
       "samples_per_scan",
       "scan_frequency_measured",
       "status_flags",
       "load_indication",
       "device_family",
       "mac_address",
       "hmi_display_mode",
       "hmi_language",
       "hmi_button_lock",
       "hmi_parameter_lock",
       "ip_mode_current",
       "ip_address_current",
       "subnet_mask_current",
       "gateway_current",
       "system_time_raw",
       "user_tag",
       "user_notes",
       "locator_indication",
       ],
       "error_code":0,
       "error_text":"success"
       }
```

2.2.2 get_parameter-读取参数

命令get_parameter读取一个或多个参数的当前值：
http://< sensor IP address >/cmd/get_parameter?list =< param1 >;< param2 >
命令参数
list-分号分隔的参数名称列表（可选）
如果未指定参数列表，则命令将返回所有可用参数的当前值。
例
Query:http://< sensor IP address >/cmd/get_parameter?list = scan_frequency;scan_frequency_measured

```
Reply: {
       "scan_frequency":50,
       "scan_frequency_measured":49.900000,
       "error_code":0,
       "error_text":"success"
       }
```

2.2.3 set_parameter-更改参数

使用命令set_parameter，可以更改任何写可访问参数的值：

http://< sensor IP address >/cmd/set_parameter?< param1 >=< value > & < param2 >=< value >

请注意：

如果指定为命令参数的任何参数未知或只读参数，则命令set_parameter返回错误消息。 在这种情况下，返回值error_code和error_text具有适当的值（见第 1.2.6 节）。

命令参数

<param1> = <value> – new <value> for parameter <param1>

<param2> = <value> – new <value> for parameter <param2>

Example

Query: http://< sensor IP address >/cmd/set_parameter?scan_frequency = 50

```
Reply: {
       "error_code":0,
       "error_text":"success"
       }
```

2.2.4 reset_parameter-将参数重置为其默认值

命令reset_parameter将一个或多个参数重置为出厂默认值：

http://< sensor IP address >/cmd/reset_parameter?list =< param1 >;< param2 >

命令参数

list-分号分隔的参数名称列表（可选）

请注意：

如果未指定参数列表，则命令将为所有可使用set_parameter写入的参数加载出厂默认值！

请注意：

此命令适用于仅通过set_parameter命令访问的全局 R / W 参数。如果参数列表包含未知或只读参数，将返回错误消息。

请注意：

将参数重置为其默认值可能需要重新启动设备才能生效。 例如，这适用于所有以太网配置参数（参见第 2.5 节）。

例子

Query: http://< sensor IP address >/cmd/reset_parameter?list = scan_frequency;scan_direction

```
Reply: {
       "error_code":0,
       "error_text":"success"
       }
```

2.2.5 reboot_device-重新启动传感器固件

命令reboot_device会触发传感器固件的软重启：

http://< sensor IP address >/cmd/reboot_device
命令参数
该命令不接受附加参数。重新启动在发送 HTTP 回复后不久执行。
请注意：
重新启动会终止所有正在运行的扫描数据输出。所有扫描数据句柄都无效，必须在重新启动后从头更新（见第 3.4 节）。
请注意：
设备重新启动最多需要 60 秒（取决于传感器配置）。当传感器再次应答 HTTP 命令请求并且系统状态标志初始化（见第 2.8.2 节）被清除时，重新启动完成。

例

Query: http://< sensor IP address >/cmd/reboot_device

Reply: {

```
"error_code":0,
"error_text":"success"
}
```

2.2.6 factory_reset-将传感器重置为出厂设置

协议版本 1.01 添加了命令factory_reset，该命令执行将所有传感器设置完全重置为出厂默认值并重新启动设备。它的结果类似于没有任何参数的reset_parameter调用，后面是对reboot_device的调用。
命令参数
该命令不接受附加参数。在发送 HTTP 回复后不久，就会执行出厂重置和设备重新启动。
例

Query: http://< sensor IP address >/cmd/factory_reset

Reply: {

```
"error_code":0,
"error_text":"success"
}
```

2.3 基本传感器信息

本节介绍用户可用的所有传感器参数。

2.3.1 参数概述

下表列出了提供基本传感器信息的大量参数（大多为只读）：

parameter name	type	description	access
device_family	uint	Numeric unique identifier (see below)	sRO
vendor	string	Vendor name (max. 100 chars)	sRO
product	string	Product name (max. 100 chars)	sRO
part	string	Part number (max. 32 chars)	sRO
serial	string	Serial number (max. 32 chars)	sRO
revision_fw	string	Firmware revision (max. 32 chars)	sRO
revision_hw	string	Hardware revision (max. 32 chars)	sRO
user_tag	string	User defined name (max. 32 chars)	RW
user_notes	string	User notes (max. 1000 Byte)	RW

这些条目与 IO-Link 设备上可用的通用信息相当。与 IO-Link 相反，大多数字符串没有大小限制。此外，每个参数可以使用命令get_parameter单独读取。

2.3.2 设备系列（device_family）

参数device_family可用于标识兼容的设备系列。单个设备系列被定义为具有相同功能（关于以太网协议）的设备组。此标识符可用于检查连接的设备是否与客户端应用程序兼容（例如 DTM 用户界面）。
目前为device_family定义了以下值：

value	description
0	reserved
1	OMDxxx-R2000 (UHD raw data devices)
2	reserved
3	OMDxxx-R2000-HD (HD raw data devices)

2.3.3 用户定义的字符串(user_tag，user_notes)

参数user_tag和user_notes是字符串，可以由用户使用没有限制（除了有效的 UTF-8 编码 - 参见 2.1 节中的类型字符串的定义）。user_tag的默认值通常是产品名称（参数产品）的简短版本，而user_notes在默认情况下为空。

2.4 传感器能力

2.4.1 参数概述

以下静态只读参数描述传感器功能：

parameter name	type	unit	description	access
max_connections	uint		max. number of scan data channels (client connections)	sRO
feature_flags	array		sensor feature flags (see below)	sRO
radial_range_min	double	m	min. measuring range (distance)	sRO
radial_range_max	double	m	max. measuring range (distance)	sRO
radial_resolution	double	m	max. radial resolution (distance)	sRO
angular_fov	double	°	max. angular field of view	sRO
angular_resolution	double	°	max. angular resolution	sRO
scan_frequency_min	uint	Hz	min. supported scan frequency (see section 2.6)	sRO
scan_frequency_max	uint	Hz	max. supported scan frequency (see section 2.6)	sRO
sampling_rate_min	uint	Hz	min. supported sampling rate (see section 2.6)	sRO
sampling_rate_max	uint	Hz	max. supported sampling rate (see section 2.6)	sRO
max_scan_sectors	uint		max. number of scan sectors (0 if not supported)	sRO
max_data_regions	uint		max. number of data regions (per scan data channel)	sRO

2.4.2 设备特征（feature_flags）

参数feature_flags返回可用于查询设备的 JSON [9]编码特征列表。目前定义了以下功能：

feature name	description	reference
ethernet	Ethernet interface	

如果某个功能可用，其名称将列在feature_flags数组中。

2.5 以太网配置

2.5.1 参数概述

以下参数允许以太网接口的配置更改：

parameter	type	description	access	default
ip_mode	enum	IP address mode: static, dhcp, autoip	RW	static
ip_address	ipv4	static IP mode: sensor IP address	RW	10.0.10.9
subnet_mask	ipv4	static IP mode: subnet mask	RW	255.255.255.0
gateway	ipv4	static IP mode: gateway address	RW	0.0.0.0
ip_mode_current	enum	current IP address mode: static, dhcp, autoip	RO	static
ip_address_current	ipv4	current sensor IP address	RO	10.0.10.9
subnet_mask_current	ipv4	current subnet mask	RO	255.255.255.0
gateway_current	ipv4	current gateway address	RO	0.0.0.0
mac_address	string	sensor MAC address (00-0D-81-xx-xx-xx)	RO	–

只读参数ip_mode_current，ip_address_current，subnet_mask_current和gateway_current提供对当前活动 IP 配置的访问。当通过 DHCP 或AutoIP使用自动 IP 配置时，这是特别有用的。
请注意：
对以太网配置的任何更改（使用set_parameter或reset_parameter）仅在系统重新启动后应用！命令reboot_device（请参阅第 2.2.5 节）可用于使用以太网协议启动重新启动。

2.5.2 IP 地址模式（ip_mode）

参数ip_mode配置以下 IP 地址模式之一：
static - IP 配置使用ip_address，subnet_mask，gateway

autoip - 使用“零配置网络”的自动 IP 配置[13]
dhcp - 使用 DHCP 服务器的自动 IP 配置
请注意：
使用 DHCP 或AutoIP的自动 IP 配置，如果尚未向传感器分配有效的 IP 地址（例如，如果未找到 DHCP 服务器），则参数ip_address_current和subnet_mask_current可能返回无效的 IP 地址 0.0.0.0。

2.6 测量配置

2.6.1 参数概述

以下（全局）参数可用于基本测量配置：

parameter name	type	unit	description	access	default
operating_mode	enum	–	mode of operation: measure, transmitter_off	vRW	measure
scan_frequency	double	1 Hz	scan frequency (10 Hz..50 Hz)	RW	35 Hz
scan_direction	enum	–	direction of rotation: cw or ccw	RW	ccw
samples_per_scan	uint	samples	number of readings per scan	RW	3600
scan_frequency_measured	double	1 Hz	measured scan frequency	RO	–

2.6.2 操作模式（operating_mode）

参数operating_mode控制传感器的操作模式。目前，可以使用以下模式：

parameter name	description
measure	Sensor is recording scan data
transmitter_off	Transmitter is disabled, no scan data is recorded

模式 measure 是传感器的正常操作模式，并且在上电后默认。模式transmitter_off允许用户去激活发射器，例如。以避免与其它光学装置的干扰。如果没有扫描数据连接是活动的，即所有句柄都已被释放，则只能执行从测量到发射机关闭的模式切换。当操作模式设置为transmitter_off时，不能请求新的扫描数据连接句柄（见第 3.2 节）。该状态也由系统状态标志scan_output_muted（见第 2.8.2 节）发出。
例如
Query: http://< sensor IP address >/cmd/set_parameter? operating_mode = measure

Reply:
```
{
    "error_code":0,
    "error_text":"success"
    }
```

2.6.3 扫描频率（scan_frequency，scan_frequency_measured）

参数scan_frequency定义传感器头转速的设置点，因此定义每秒记录的扫描次数（详见 3.1 节）。 对于 R2000 有效值，范围为 10 Hz 至 50 Hz，步长为 1 Hz（默认值为 35 Hz）。非整数值自动四舍五入为整数值。参数scan_frequency_measured以 0.1 Hz 的分辨率读取传感器头

的旋转速度的实际值。它是只读参数。

例如

Query: http://< sensor IP address >/cmd/get_parameter?list = scan_frequency;scan_frequency_measured

```
Reply: {
       "scan_frequency":35,
       "scan_frequency_measured":34.900000,
       "error_code":0,
       "error_text":"success"
       }
```

表 2.1：参数samples_per_scan（R2000 UHD）的有效值列表

samples per scan	angular resolution	maximum scan frequency	sampling rate (maximum)
25 200	0.014°	10 Hz	252 kHz
16 800	0.021°	15 Hz	252 kHz
12 600	0.029°	20 Hz	252 kHz
10 080	0.036°	25 Hz	252 kHz
8400	0.043°	30 Hz	252 kHz
7200	0.050°	35 Hz	252 kHz
6300	0.057°	40 Hz	252 kHz
5600	0.064°	45 Hz	252 kHz
5040	0.071°	50 Hz	252 kHz
4200	0.086°	50 Hz	210 kHz
3600	0.100°	50 Hz	180 kHz
2400	0.150°	50 Hz	120 kHz
1800	0.200°	50 Hz	90 kHz
1440	0.250°	50 Hz	72 kHz
1200	0.300°	50 Hz	60 kHz
900	0.400°	50 Hz	45 kHz
800	0.450°	50 Hz	40 kHz
720	0.500°	50 Hz	36 kHz
600	0.600°	50 Hz	30 kHz
480	0.750°	50 Hz	24 kHz
450	0.800°	50 Hz	23 kHz
400	0.900°	50 Hz	20 kHz
360	1.000°	50 Hz	18 kHz
240	1.500°	50 Hz	12 kHz
180	2.000°	50 Hz	9 kHz
144	2.500°	50 Hz	8 kHz
120	3.000°	50 Hz	6 kHz
90	4.000°	50 Hz	5 kHz
72	5.000°	50 Hz	4 kHz

2.6.4 扫描方向（scan_direction）

参数scan_direction定义传感器头的旋转方向。用户应用程序可以选择顺时针旋转（cw）或逆时针旋转（ccw）。关于这些设置如何与传感器坐标系和扫描数据输出相关，请参见 3.1.1 和 3.1.2 节。

例如

Query: http://< sensor IP address >/cmd/set_parameter?scan_direction = ccw

```
Reply: {
       "error_code":0,
       "error_text":"success"
       }
```

2.6.5 扫描分辨率（samples_per_scan）

参数samples_per_scan定义在传感器头旋转一周期间记录的样本数量（详见 3.1 节）。目前，仅支持许多离散值。表 2.1 列出了 R2000 的所有有效值。请求每个扫描的任何其他数量的样本结果为错误消息。

请注意：

samples_per_scan乘以scan_frequency的数目给出了采样率，即每秒的测量次数。传感器支持

在 sampling_rate_min 和 sampling_rate_max 之间的采样率（见第 2.4 节）。因此，samples_per_scan的间接数也限制参数scan_frequency的最大值（反之亦然）。 因此，表 2.1 也表示最大扫描频率。

例如

Query: http://< sensor IP address >/cmd/set_parameter? samples_per_scan = 3600

Reply: {

```
"error_code":0,
"error_text":"success"
}
```

2.7 HMI /显示配置

2.7.1 参数概述

本节列出了可用于配置传感器人机界面（HMI）的所有（全局）参数，包括头显示 LED 显示屏和两个按钮。

显示模式参数允许客户端应用程序直接访问 HMI LED 显示屏。有关使用这些功能的详细说明，请参阅第 4 章。

parameter name	type	description	access	default
hmi_display_mode	enum	normal operation display mode	RW	static_logo
hmi_language	enum	display language: english, german	RW	english
hmi_button_lock	bool	lock HMI buttons: on / off	RW	off
hmi_parameter_lock	bool	set HMI to read-only: on / off	RW	off
locator_indication	bool	LED locator indication: on / off	RW	off
display mode parameters (see chapter 4)				
hmi_static_bitmap	binary	bitmap image for mode static_logo	RW	P+F logo
hmi_static_text_1	string	text line 1 for mode static_text (max. 30 chars)	RW	"Pepperl+Fuchs"
hmi_static_text_2	string	text line 2 for mode static_text (max. 30 chars)	RW	"R2000"
hmi_application_bitmap	binary	bitmap image for mode application_bitmap	vRW	<empty>
hmi_application_text_1	string	text line 1 for application_text (max. 30 chars)	vRW	<empty>
hmi_application_text_2	string	text line 2 for application_text (max. 30 chars)	vRW	<empty>

2.7.2 HMI 显示模式（hmi_display_mode）

参数hmi_display_mode在正常操作期间控制 HMI LED 显示器的内容，即，既不显示警告也不显示错误，并且用户没有激活 HMI 菜单。根据设备系列，可以使用以下显示模式：

Display mode	Device family	Description
off	all	Display is blank.
static_logo	all	Show a static logo.
static_text	all	Show two lines of static text.
bargraph_distance	OMDxxx-R2000	Show visualization of measured distances.
bargraph_echo	OMDxxx-R2000	Show visualization of measured echo values.
bargraph_reflector	OMDxxx-R2000	Show visualization of high echo targets.
application_bitmap	OMDxxx-R2000	Show an application-provided bitmap.
application_text	OMDxxx-R2000	Show two lines of application-provided text.

hmi_display_mode的设置被存储到非易失性存储器中，即，在电源周期期间保持。

2.7.3 HMI 显示语言（hmi_language）

参数hmi_language控制 HMI LED 显示屏显示的文本消息（菜单，警告，错误）的语言。目前设置英语和德语可用。当前设置存储在非易失性存储器中，即在电源周期期间保存。

2.7.4 HMI 按钮锁定（hmi_button_lock）

布尔参数hmi_button_lock允许禁用传感器前面的 HMI 按钮。如果设置为 on，则忽略任何按钮的按钮。这使客户端应用程序能够拒绝用户访问传感器的 HMI 菜单。
请注意，锁定按钮还会阻止访问只读信息，例如当前的以太网配置。如果这不是想要考虑使用参数hmi_parameter_lock代替。

2.7.5 HMI 参数锁定（hmi_parameter_lock）

协议版本 1.01 添加了布尔参数hmi_parameter_lock，允许通过传感器的 HMI 显示菜单禁用参数更改。这使得客户端应用程序能够防止用户更改传感器参数，同时保留从 HMI 菜单（例如当前以太网配置）确定可用参数的当前设置的可能性。

2.7.6 定位器指示（locator_indication）

参数locator_indication临时激活电源和 Q2 LED 的特殊闪烁模式。如果安装了多个设备，此功能可用于识别特定的 R2000 设备。定位器指示功能在重新启动，重新通电和恢复出厂设置后自动禁用。

2.8 系统状态

2.8.1 参数概述

可以访问以下（只读）参数以从传感器获取状态信息。
例如
Query: http://< sensor IP address >/cmd/get_parameter?list = up_time; power_cycles

Reply: {

```
"up_time":44,
"power_cycles":22,
"error_code":0,
"error_text":"success"
}
```

parameter name	type	unit	description	access
status information				
status_flags	bitfield	–	sensor status flags (see section 2.8.2)	RO
load_indication	uint	%	current system load (0 %... 100 %)	RO
time information				
system_time_raw	ntp64	–	raw system time (NTP timestamp format)	RO
up_time	uint	min	time since power-on	RO
power_cycles	uint	–	number of power cycles	RO
operation_time	uint	min	overall operating time	RO
operation_time_scaled	uint	min	overall operating time scaled by temperature	RO
operating conditions				
temperature_current	int	°C	current operating temperature	RO
temperature_min	int	°C	minimum lifetime operating temperature (power-up update delay 15min)	RO
temperature_max	int	°C	maximum lifetime operating temperature (power-up update delay 15min)	RO
contamination	uint	%	contamination of sensor cover **Note:** This parameter currently returns always zero.	RO

2.8.2 系统状态标志（status_flags）

只读参数status_flags（参见第 2.8 节）提供了一个系统状态标志数组：

bit	flag name	description
Generic		
0	initialization	System is initializing, valid scan data not available yet
2	scan_output_muted	Scan data output is muted by current system configuration (see section 2.6.2)
3	unstable_rotation	Current scan frequency does not match set value
4	skipped_packets	Preceding scan data packets have been skipped due to connection issues. This might occur of the client does not receive data fast enough.
Warnings		
8	device_warning	Accumulative flag – set if device displays any warning
10	low_temperature_warning	Current device-internal temperature below warning threshold (0 °C)
11	high_temperature_warning	Current device-internal temperature above warning threshold (80 °C)
12	device_overload	Overload warning – sensor CPU overload is imminent
Errors		
16	device_error	Accumulative flag – set if device displays any error
18	low_temperature_error	Current device-internal temperature below error threshold (−10 °C)
19	high_temperature_error	Current device-internal temperature above error threshold (85 °C)
20	device_overload	Overload error – sensor CPU is in overload state
Defects		
30	device_defect	Accumulative flag – set if device detected an unrecoverable defect

系统状态标志与扫描数据头状态标志（见第 3.4.3 节）类似，但提供有关当前设备状态的最新信息（与特定扫描数据不相关）。

请注意：

上表中未列出的所有标志均被保留，应被忽略。

2.8.3 系统负载指示（load_indication）

状态变量load_indication给出传感器当前 CPU 负载的粗略指示：

0％ - 系统空闲

如果 HMI 显示被禁用（hmi_display_mode == 0）并且没有活动的扫描数据输出运行（TCP 和 UDP），则系统处于空闲状态。

100％ - 系统过载

如果太多客户端通过活动的 TCP / UDP 连接请求扫描数据，则系统可能进入过载。 在这种情况下，R2000 的标称操作不能保证！请通过禁用 HMI 显示，减少 TCP / UDP 连接数或降低扫描分辨率来减少系统负载。

3 使用 TCP 或 UDP 扫描数据输出

3.1 扫描数据采集原理

R2000 是一款激光扫描仪，用于定期测量 360° 范围内的距离。在以由参数scan_frequency（参见第 2.6 节）定义的恒定频率旋转时的视场。将测量结果合并到扫描中。单次扫描对应于传感器头的一次旋转，并且产生扫描点（也称为样本）的序列。扫描中的扫描点数由参数samples_per_scan定义（参见第 2.6 节）。
每个扫描点包括对应角度的距离值以及回波振幅。然而，由于以均匀的角分辨率（取决于参数samples_per_scan）执行测量，所以实际的扫描数据输出通常仅给出每个样本的距离和幅度数据。可以通过从扫描的起始角度累加角增量来重建相应的角度读数。
扫描数据的输出格式取决于所使用的扫描数据包类型 - 有关详细信息，请参阅第 3.4 节。以下小节描述了 R2000 使用的扫描数据表示的各种基本概念。

3.1.1 传感器坐标系

传感器坐标系被定义为右手笛卡尔坐标系。图 3.1 显示了传感器顶视图和一个侧视图的坐标系：原点位于旋转轴和激光束轴的交点。X 轴指向传感器前端 - 与传感器后端的连接插孔相对。Y 轴垂直于 X 轴并且平行于传感器的基板（在图 3.1（a）中向上指向）。Z 轴与旋转轴线共线（在图 3.1（b）中指向上）。

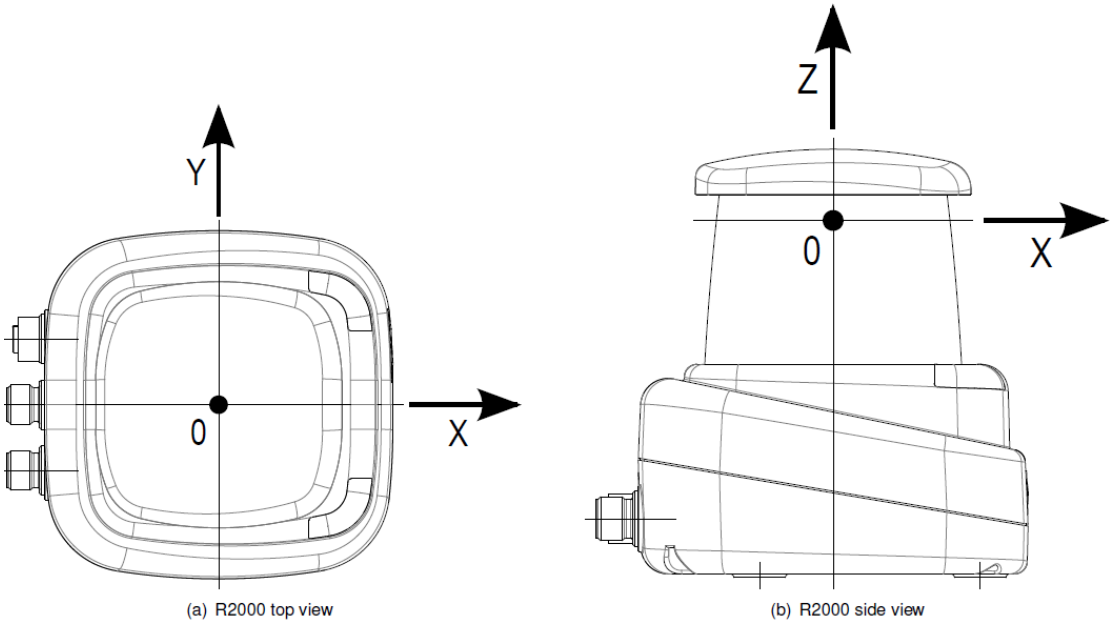

图 3.1：传感器坐标系

3.1.2 扫描数据坐标系

由传感器坐标系的 X 轴和 Y 轴形成的平面称为扫描平面。激光扫描仪的所有测量都记录在该平面内。在围绕扫描平面的原点的头旋转的方向上顺序地执行扫描数据采集。因此，扫描

数据通常在极坐标系内表示（参见图 3.2（a））。坐标系的极点由旋转轴（传感器坐标系的 Z 轴）定义。角度测量（极轴）的参考等于传感器坐标系的 X 轴（在图 3.2（a）指向上）。
在标称操作期间，使用均匀的角增量和旋转方向连续记录扫描点。可以通过全局设备参数（参见第 2.6 节）配置角增量和旋转方向。默认情况下，激光扫描器以数学正方向旋转。该方向被称为逆时针（缩写为ccw）-两个连续扫描点之间的角增量具有正值。相反的方向因此被称为顺时针（缩写为cw）-两个连续扫描点之间的角增量具有负值。
图 3.2（b）显示了一个角度增量为 7.5° 的激光扫描的例子。参考极轴在扫描平面内计算单个扫描点（角坐标）的测量角度。测量距离（径向坐标）由从旋转中心（极点）到被激光束照射的物体的距离确定。 角坐标在 360° 内视野的值范围为[-180°，180°)，[-180° 包括-180° 但不包括 180° 。

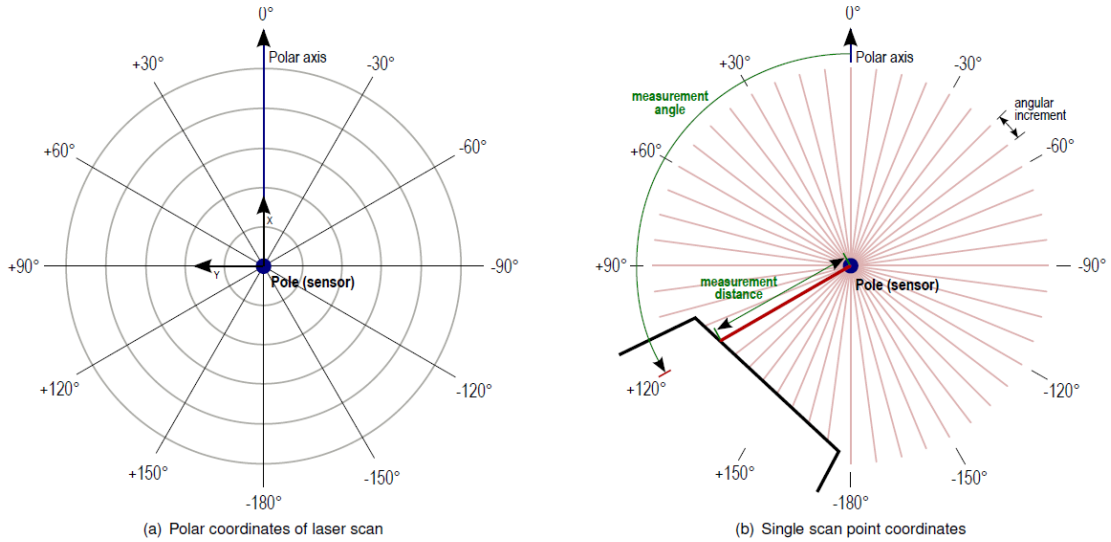

图 3.2：传感器扫描平面内的激光扫描

3.1.3 距离读数

距离读数通常作为扫描数据包类型定义的整数值输出（见第 3.4 节）。在无效测量（例如，没有检测到回波或距离超出范围）的情况下，距离读数被设置为误差替换值：距离值的最大可表示整数值（例如对于 uint32 类型的距离值的 0xFFFFFFFF）。
请注意：
测量分辨率和测量范围受传感器数据表中列出的传感器的物理能力限制。该信息也可通过只读变量radial_resolution，radial_range_min和radial_range_max获得（参见 2.4 节）。

3.1.4 回波强度读数

对于传感器的每个测量，可选的强度数据可用于客户端。幅度数据作为无量纲线性值输出，具有 12 位的固定分辨率。
原则上，强度数据可以仅传送对象的相对反射率的估计。测量的强度取决于目标对象的表面特性（其绝对反射率），其到传感器的距离，传感器激光束在目标表面上的入射角等等-因此，强度数据的直接比较仅是可行的 对于在相似的观察条件下的物体表面。
请注意：

请注意，强度数据未校准。因此，即使在类似的观察条件下，不同传感器装置的幅度数据也可能不相同！
12 位强度数据的最低有效值保留用于以下特殊值：

value	name	description
0	no echo	receiver detected no echo
1	blinding	receiver overloaded due to excessive echo amplitude
2	error	unable to measure amplitude
3-31	reserved	reserved for internal use
>31	amplitude	measured echo amplitude value

3 到 31 范围内的所有值都保留供内部使用。最小的实际幅度值（实际测量的）为 32。

3.1.5 时间戳

扫描数据输出包括两种类型的时间戳：
原始时间戳
原始时间戳由在上电时从零开始计数的内部系统时钟生成。它的分辨率是优于 1 ms，漂移低于 100 ppm。原始时间总是向前计数，没有任何不连续或溢出。
同步时间戳
同步时间戳记使用与外部时间服务器同步的时钟源。请注意，这些时间戳可能受到由于时间服务器同步所导致的不连续性的影响。时间戳总是以 64 位 NTP 时间戳格式提供（详见[2]）。原始系统时间可通过设备参数system_time_raw访问（参见第 2.8 节）。当使用get_parameter读取system_time_raw时，设备将返回接收到命令时的时间点的原始系统时间。请注意，发送时间戳请求和接收带有时间戳的应答都受到非确定性 HTTP 传输延迟的影响。
请注意：
此功能尚未完全实现！目前，同步时间戳不可用，并输出为零。

3.2 扫描数据输出原理

3.2.1 介绍

为了从激光扫描器接收扫描数据，客户端应用需要建立到传感器的扫描数据连接。基本上激光扫描器支持两种不同类型的数据通道：TCP 和 UDP。TCP 信道以潜在不可预测的等待时间为代价提供用于扫描数据的传输的安全和防错的信道。相比之下，UDP 通道允许以最小延迟进行数据传输，同时牺牲潜在的不可恢复的数据损坏或数据丢失。使用 HTTP 命令接口管理 TCP 和 UDP 数据通道。
对于典型应用，需要以下步骤来使用扫描数据输出：
1.如有必要，设置扫描仪的全局配置（参见第 2 章）
2.建立传感器的数据通道（见 3.3.1 节和 3.3.2 节）
3.如有必要，配置扫描数据输出（见第 3.3.6 节）
4.开始扫描数据传输（见 3.3.4 节）
5.从设备接收扫描数据（见第 3.4 节）

6.停止扫描数据传输（见 3.3.5 节）
7.终止传感器的数据通道（见第 3.3.3 节）
第 3.3 节详细介绍了管理扫描数据输出所需的命令。

3.2.2 扫描数据连接句柄

R2000 支持到多个客户端的并行扫描数据连接。为了配置单独控制这些连接，每个连接由唯一的连接句柄标识。句柄被定义为最多 16 个字符的随机字母数字字符串。传感器确保每个手柄仅用于一个活动扫描数据连接。应用程序不应该对句柄的结构做任何进一步的假设，因为实现细节可能会随着新的固件版本（另见下文）而改变。
兼容性来处理 R2000 固件 v1.0x 的实现
自第一版本的以太网通信协议以来，连接句柄已被指定为随机的字母数字字符串。不幸的是，v1.20 之前的 R2000 固件版本实现了相当系统的句柄生成算法，导致后续句柄接收线性增加的签名。使用固件 v1.20，此实现已更新为随机的句柄生成模式。
此实现更改应该不会导致大多数客户端应用程序的问题。然而，如果客户端应用程序依赖于固件 v1.0x 的特定句柄生成模式，则需要修改它以使用固件 v1.20 或更高版本的新实现。出于向后兼容性的原因，当前固件版本提供了恢复旧句柄生成模式的选项。要激活此选项，需要使用命令set_parameter将布尔值参数deprecated_handle_generation设置为 on。该参数存储在非易失性存储器中，即，它只需要设置一次。其默认值为 off。
请注意：
将在以后的固件版本中删除参数deprecated_handle_generation。强烈建议将客户端应用程序适配到更新的句柄生成模式。

3.2.3 扫描数据连接看门狗

默认情况下，每个扫描数据连接（由句柄标识）具有一个看门狗定时器。如果在定义的时间段内未使用数据通道，则相关的扫描数据输出将停止，数据通道将关闭，数据通道手柄将被传感器无效。这样，设备可以为新的扫描数据连接释放宝贵的资源，否则这些连接将被“僵尸”连接永久阻止。
为了防止看门狗超时，客户端需要定期馈送看门狗。这可以通过命令feed_watchdog（参见第 3.3.8 节）或者使用用于 TCP 扫描数据连接的“在线”看门狗进给来完成（见第 3.3.9 节）。每次呼叫都会重置看门狗定时器。
可以使用命令request_handle_udp（请参阅第 3.3.1 节），request_handle_tcp（参见第 3.3.2 节）和set_scanoutput_config（参见第 3.3.6 节）中的参数 watchdog 和watchdogtimeout为客户端应用程序单独为每个扫描数据连接配置看门狗超时时间。参数watchdogtimeout指定超时时间，范围为 1 秒到 500 秒。参数watchdog使能（值 on）或禁用（值 off）看门狗。默认情况下，看门狗使能，超时周期为 60 秒。
请注意：
虽然看门狗超时周期（看门狗超时）可以指定为 1 ms的分辨率，但当前固件版本使用的有效内部分辨率约为 10 s。客户端软件不应该依赖更短的反应时间。
请注意：
启用看门狗（watchdog）或更改看门狗超时周期（watchdogtimeout）使用命令set_scanoutput_config会隐式地向看门狗发送数据（即复位看门狗超时）。

3.2.4 扫描数据输出自定义

客户端可以定制在扫描数据通道上如何输出扫描数据的一些属性。这些配置设置特定于由唯一连接句柄标识的单个数据连接。使用request_handle_tcp（请参阅第 3.3.2 节）和request_handle_udp（参见第 3.3.1 节）启动新的扫描数据连接时，可以设置这些设置，也可以使用set_scanoutput_config（参见第 3.3.6 节）更改现有的扫描数据连接。

选择起始角度

客户端可以使用参数start_angle（参见第 3.3.6 节）配置扫描（索引 0）内的第一扫描点的（极坐标）角度。默认情况下，它设置为-180.0°（即，start_angle =-1800000）。扫描数据流中的后续扫描点（索引（n + 1））根据当前传感器头旋转方向排序。start_angle的值指的是相对于扫描数据坐标系的测量角度（见第 3.1.2 节）。

限制扫描点数

协议版本 1.01 将参数max_num_points_scan添加到扫描数据连接的可自定义选项列表。此参数允许限制通过扫描数据连接输出的扫描点数。与全局参数samples_per_scan（其控制传感器记录每次扫描的样本数）相反，max_num_points_scan的设置仅影响为特定扫描数据连接输出的扫描点数。该参数指定为无符号整数（uint），并接受任何非负数。值 0 被识别为“无限制”的特殊情况，即传感器总是输出所有扫描点。这也是默认值。如果max_num_points_scan设置为samples_per_scan以下的值，则客户端应用程序接收到的扫描点比传感器记录少。结合参数start_angle，这允许客户端应用程序仅获取扫描的段（扇区），而不是所有记录的扫描点。图 3.3 显示了这样的设置。这可以非常有用的减少数据流量如果完整的 360° 的传感器的视角不需要的话。

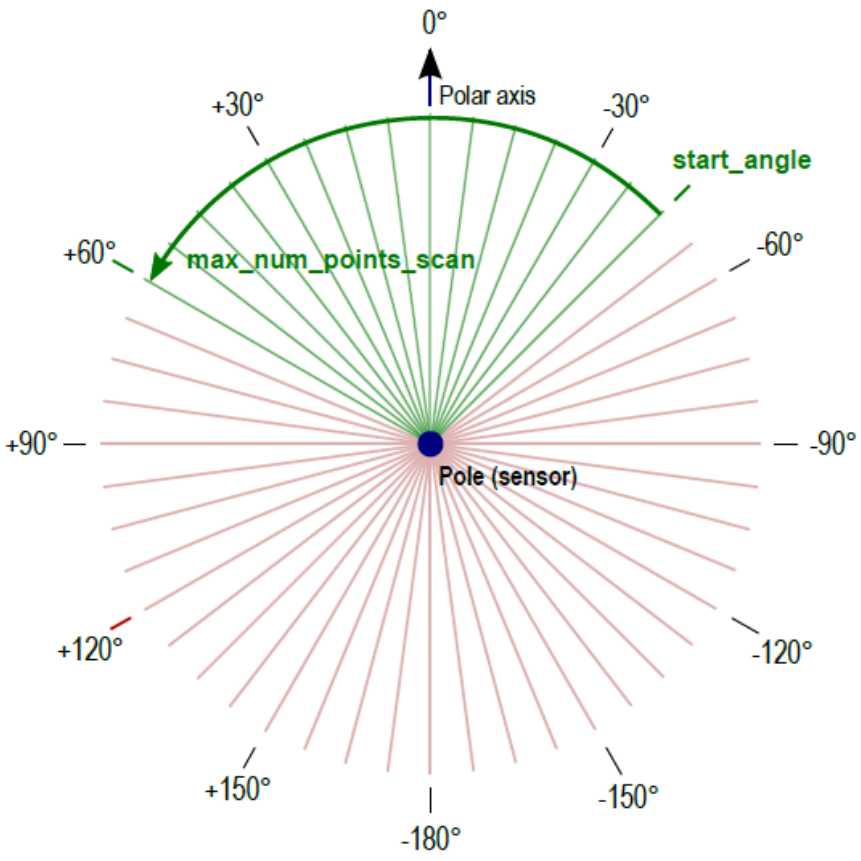

图 3.3：对段的扫描输出的限制

3.2.5 使用多个并发扫描数据连接

如前所述，扫描数据协议被设计为支持多个并发扫描数据连接。但是，当前 R2000 设备的 CPU 资源有限。这取决于传感器的测量配置（见第 2.6 节），可以运行多少并发连接，而不会由于系统过载而产生不良影响。
客户端应用软件的责任是确保传感器可以处理来自并行数据通道的系统负载(见第 1.1 节)。连接句柄的成功请求（参见第 3.3.1 和 3.3.2 节）和成功的连接建立都不能保证传感器可以实时连续提供所请求的数据量。通过读取系统状态变量load_indication（参见第 2.8.8 节）可以监控设备的系统负载。
最大连接数限制为max_connections的值（见第 2.4 节）。
请注意：
单个客户端数据通道可以由传感器处理而没有任何限制。

3.3 用于管理扫描数据输出的命令

后续部分描述所有命令。

3.3.1 request_handle_udp-请求基于 UDP 的扫描数据通道

命令request_handle_udp用于请求从传感器到客户端(数据的传输)的基于 UDP 的扫描数据传输的句柄。 如果成功，传感器将使用在句柄请求处指定的目标 IP 地址和 UDP 端口将扫描数据发送到客户端。 图 3.4 给出了当使用基于 UDP 的通道进行扫描数据输出时传感器和客户端之间通信的概述。
命令参数
命令request_handle_udp接受以下参数：

argument name	type	unit	description	default
address	ipv4	–	required: target IP address of the client	–
port	uint	–	required: target port for UDP data channel (client side)	–
watchdog	bool	on / off	optional: enable or disable connection watchdog	on
watchdogtimeout	uint	1 ms	optional: connection watchdog timeout period	60 000 ms
packet_type	enum	–	optional: scan data packet type (see section 3.4)	A
start_angle	int	1/10 000°	optional: angle of first scan point for scan data output	-1800000
max_num_points_scan	uint	samples	optional: limit number of points in scan data output	0 (unlimited)

请注意，request_handle_udp可选地接受set_scanoutput_config命令的所有参数（请参见第 3.3.6 节）-除了参数句柄。这样，可以在数据通道启动期间指定扫描数据输出的初始配置。
命令返回值
句柄 - 唯一（随机）字母数字字符串作为新 UDP 数据通道的标识符（句柄）
在有效的命令调用期间，扫描器使用指定的目标 IP 地址和端口号向客户端创建一个新的 UDP 通道。扫描器可以拒绝创建新的 UDP 信道的请求，例如如果超过并发客户端连接的最大数量。如果发生错误，句柄的返回值无效，error_code / error_text返回有关否定响应原因的详细信息（请参见第 1.2.6 节）。
请注意：由于从传感器到客户端建立了 UDP 扫描数据连接（“传入连接”），因此它很容易被防火墙软件阻止。请确保您的防火墙设置允许从传感器 IP 地址到客户端应用程序的传入 UDP 连接。

请注意：应用程序不应对句柄的结构做任何假设。句柄应被视为最大的随机字母数字字符串。16 个字符。

命令示例

Query: http://<sensor IP address>/cmd/request_handle_udp?address=192.168.10.20&port=54321&packet_type=C

Reply: {

```
"handle":"s10",
"error_code":0,
"error_text":"success"
}
```

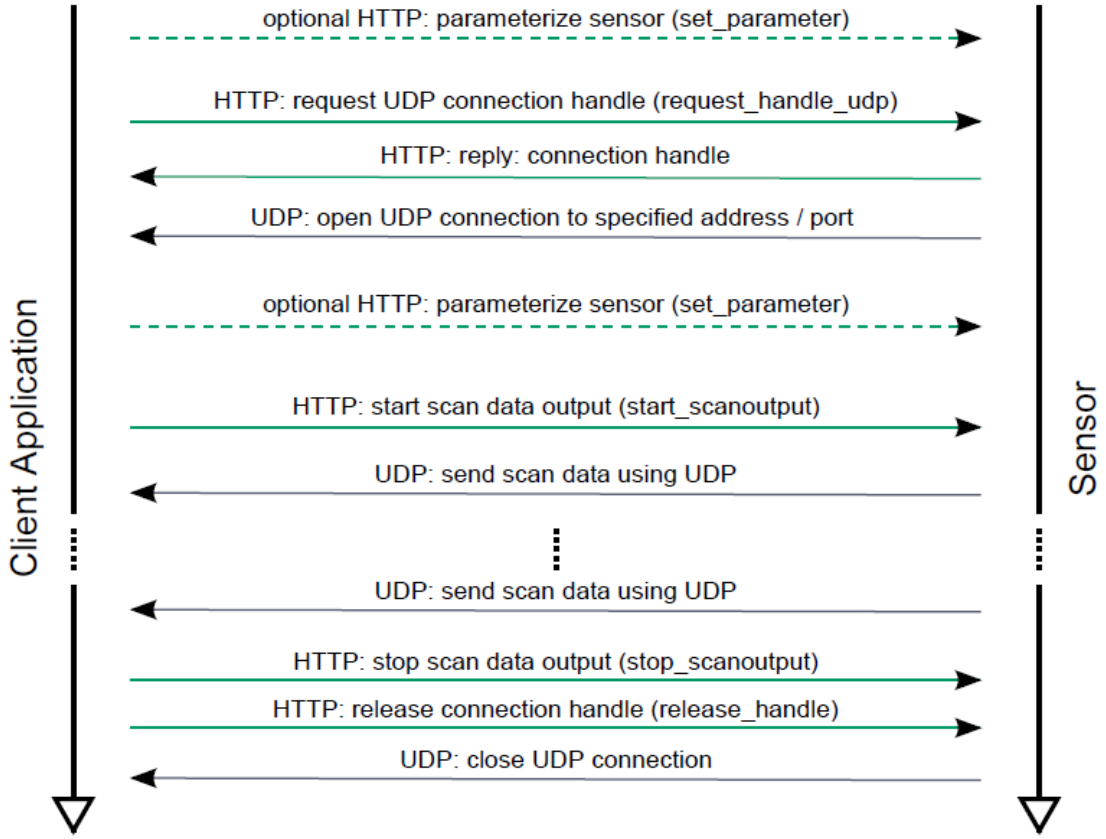

图 3.4：时间轴：使用 UDP 扫描数据传输

3.3.2 request_handle_tcp-请求基于 TCP 的扫描数据通道

命令request_handle_tcp用于请求从传感器到客户端的基于 TCP 的扫描数据传输的句柄。如果成功，则允许客户端创建到传感器的新的 TCP 连接，以便接收扫描数据。图 3.5 给出当使用基于 TCP 的信道用于扫描数据输出时传感器和客户端之间的通信的概述。

命令参数

命令request_handle_tcp接受以下参数：

argument name	type	unit	description	default
address	ipv4	–	optional: IP address of the client	(see below)
port	uint	–	optional: desired port for client connection (sensor side)	(see below)
watchdog	bool	on / off	optional: enable or disable connection watchdog	on
watchdogtimeout	uint	1 ms	optional: connection watchdog timeout period	60 000 ms
packet_type	enum	–	optional: scan data packet type (see section 3.4)	A
start_angle	int	1/10 000°	optional: angle of first scan point for scan data output	-1800000
max_num_points_scan	uint	samples	optional: limit number of points in scan data output	0 (unlimited)

请注意，request_handle_tcp可选地接受set_scanoutput_config命令的所有参数（参见第 3.3.6 节） - 除了参数句柄。这样，可以在数据通道启动期间指定扫描数据输出的初始配置。

命令返回值

句柄 - 唯一（随机）字母数字字符串作为新TCP数据通道的标识符（句柄）

端口 - 新TCP数据通道的端口号

成功时，传感器在端口中返回一个 TCP 端口号，该端口现在对客户端数据连接打开。 注意，每个端口只接受一个 TCP 连接！如果在调用request_handle_tcp时指定了参数地址，则扫描程序只接受源自给定 IP 地址的 IP 连接（使用返回的句柄）。

使用参数端口，客户端可以尝试为其 TCP 连接保留特定端口。如果此端口已在使用，request_handle_tcp返回一个错误。如果未指定参数端口，则传感器自动选择范围 32768 - 61000 内的临时端口。

对request_handle_tcp的调用可能会失败，例如 如果达到并发客户端连接的最大数量。 如果发生错误，句柄和端口的返回值无效，error_code / error_text提供错误详细信息（请参见第 1.2.6 节）。

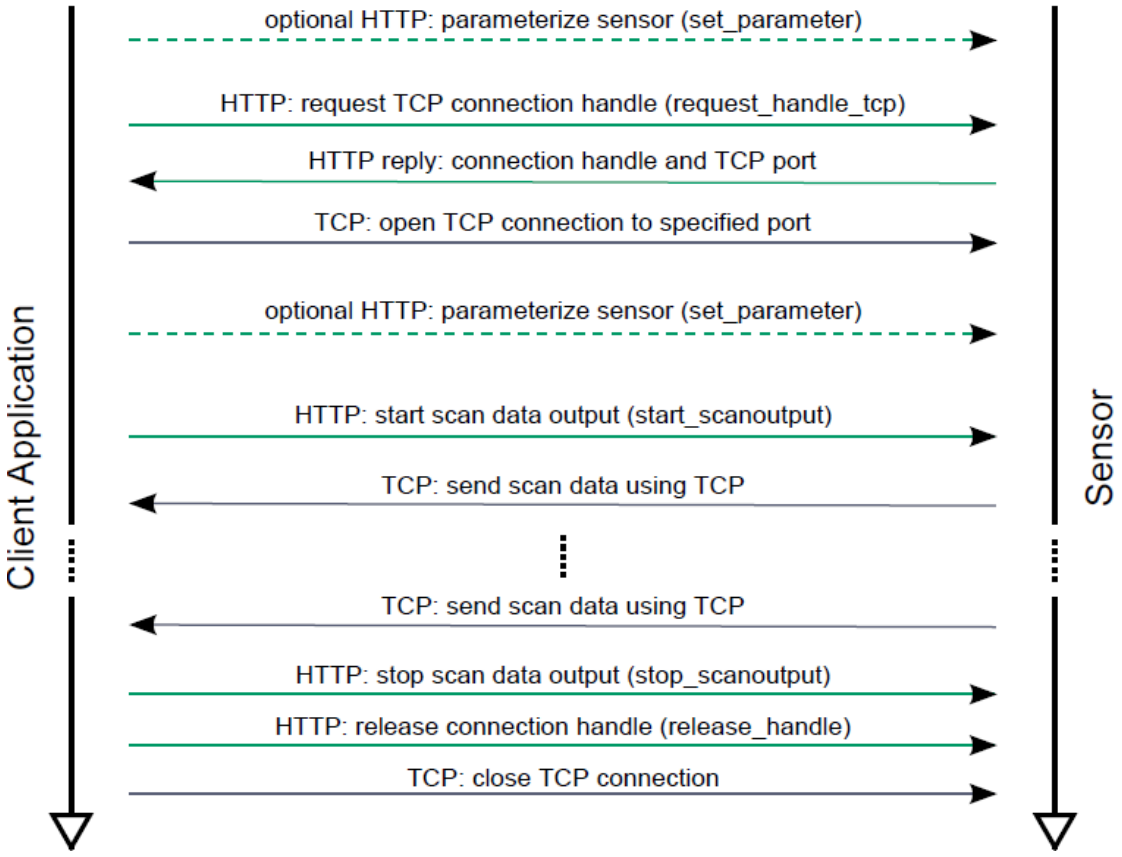

图 3.5：时间轴：使用 TCP 扫描数据传输

命令示例

http://<sensor IP address>/cmd/request_handle_tcp?packet_type=A&watchdogtimeout=1000&start_angle=0

Reply: {

```
"port":39731,
"handle":"s22",
"error_code":0,
"error_text":"success"
}
```

3.3.3 release_handle-释放数据通道句柄

使用命令release_handle，客户端可以释放数据通道句柄。使用此句柄的任何活动扫描数据输出将立即停止。相关的基于 UDP 的数据通道由传感器本身关闭。基于 TCP 的相关数据通道应由客户端关闭。

命令参数

argument name	type	description
handle	string	handle for scan data channel (max.16 chars) (required argument – always specified first)

命令实例

Query: http://< sensor IP address >/cmd/release_handle? handle = s22

Reply: {

```
"error_code":0,
"error_text":"success"
}
```

3.3.4 start_scanoutput-启动扫描数据的输出

命令start_scanoutput开始传输由给定句柄指定的数据通道的扫描数据。当开始时，传感器将使用给定句柄使用已建立的 UDP 或 TCP 通道开始向客户端发送扫描数据 - 请参见第 3.3.1 节和第 3.3.2 节。（重新）开始扫描数据传输也会重置扫描数据头中的扫描编号和扫描包编号的计数器（请参见第 3.4.2 节）。扫描数据输出始终从新扫描开始（扫描编号为 0，扫描包编号为 1）开始。

命令参数

argument name	type	description
handle	string	handle for scan data channel (max.16 chars) (required argument – always specified first)

命令实例

Query: http://< sensor IP address >/cmd/start_scanoutput? handle = s22

Reply: {

```
"error_code":0,
"error_text":"success"
}
```

3.3.5 stop_scanoutput - 终止扫描数据的输出

命令stop_scanoutput停止传输由给定句柄指定的数据通道的扫描数据。扫描数据输出立即在当前扫描数据包之后停止 - 不一定是在完整扫描结束时。

请注意：

由于 TCP 堆栈数据队列的缘故，TCP 客户端在发送stop_scanoutput后仍然可能会收到多个扫描数据包。

命令参数

argument name	type	description
handle	string	handle for scan data channel (max.16 chars) (required argument – always specified first)

命令实例

Query: http://< sensor IP address >/cmd/stop_scanoutput? handle = s22

Reply: {

```
"error_code":0,
"error_text":"success"
}
```

3.3.6 set_scanoutput_config-重新配置扫描数据输出

使用命令set_scanoutput_config，客户端可以为每个活动的扫描数据输出通道单独参数化扫描数据输出。所有命令参数仅适用于扫描数据的输出。参考记录测量（扫描数据）的（全局）参数的定制通过使用命令set_parameter（参见第 2.6 节）来完成。

命令参数

argument name	type	unit	description	default
handle	string	–	handle for scan data channel (max.16 chars) (required argument – always specified first)	–
watchdog	bool	on / off	optional: enable or disable connection watchdog	on
watchdogtimeout	uint	1 ms	optional: connection watchdog timeout period	60 000 ms
packet_type	enum	–	optional: scan data packet type: A, B, C, (see section 3.4)	A
start_angle	int	1/10 000°	optional: angle of first scan point for scan data output	-1800000
max_num_points_scan	uint	samples	optional: limit number of points in scan data output	0 (unlimited)

建议（但不是必需）在使用set_scanoutput_config时停止传感器数据输出。如果扫描数据输出处于活动状态，则修改的配置设置应用于正在运行的数据流的时间点是非确定性的。应用新设置后，将暂停扫描数据输出，直到开始新扫描（跳过扫描中间的数据分组）。如果客户端应用程序依赖于确定性切换行为，它应该首先使用stop_scanoutput停止扫描数据传输，使用set_scanoutput_config更改设置，最后使用start_scanoutput重新启动数据流。

参数start_angle

用户可以通过参数start_angle来控制扫描的第一扫描点的角度。有效值的范围是[-1800000，+1800000)包括-1800000(-180°)但不包括+1800000(180°)。指定的值不必与扫描数据采集的配置角度分辨率相匹配（参见第 2.6 节）-传感器将使用记录角度等于或大于指定旋转方向角度的第一个扫描点开始扫描数据输出。

请注意：

命令get_scanoutput_config（参见第 3.3.7 节）将返回用户指定的精确值，而扫描数据包头部

中的“第一扫描点的绝对角度”（见第 3.4.2 节）将指定第一个扫描点实际使用。
命令实例

Query: http://<sensor IP address>/cmd/set_scanoutput_config?handle=s22&packet_type=B&start_angle=-90000

Reply: {

```
"error_code":0,
"error_text":"success"
}
```

3.3.7 get_scanoutput_config -读取扫描数据输出配置

命令get_scanoutput_config返回指定扫描数据输出通道（UDP 或 TCP）的当前扫描数据输出配置。
命令参数

argument name	type	description
handle	string	handle for scan data channel (max.16 chars) (required argument – always specified first)
list	string	semicolon separated list of parameter names (optional)

如果未指定参数列表，则命令将返回所有可用配置参数的当前值（请参见第 3.3.6 节）。
命令实例

Query: http://<sensor IP address>/cmd/get_scanoutput_config?handle=s22&packet_type=B&start_angle=-900000

Reply: {

```
"address":"0.0.0.0",
"port":39050,
"watchdog":"on",
"watchdogtimeout":60000,
"packet_type":"A",
"start_angle":-1800000,
"error_code":0,
"error_text":"success"
}
```

3.3.8 feed_watchdog-喂狗连接

命令feed_watchdog馈送连接看门狗，即每次调用此命令将复位看门狗定时器。有关连接看门狗机制的详细说明，请参阅第 3.2.3 节。
命令参数

argument name	type	description
handle	string	handle for scan data channel (max.16 chars (required argument – always specified first)

命令实例

Query: http://< sensor IP address >/cmd/feed_watchdog? handle = s36924971

```
Reply: {
       "error_code":0,
       "error_text":"success"
       }
```

请注意：
使用命令set_scanoutput_config（参见第 3.3.6 节）启用看门狗（看门狗）或更改看门狗超时周期（watchdogtimeout）也隐含地向看门狗发送。

3.3.9 TCP 在线看门狗源

PFSDP 版本 1.01 增加了对使用现有双向 TCP 扫描数据连接的反向通道的 TCP“在线”看门狗源的支持（见第 3.3.2 节）。它允许馈送连接看门狗，而不是对现有feed_watchdog命令所需的每个馈送操作强加 HTTP 连接（参见第 3.3.8 节）。
进给顺序(喂狗顺序)
为了向现有 TCP 扫描数据连接的监视器馈送，需要从客户端应用程序向传感器发送以下字节序列：
0x66 0x65 0x65 0x64 0x77 0x64 0x67 0x04
此 8 字节序列表示传感器可识别的 ASCII 字符串feedwdg ＜ eot ＞。传感器不发送任何确认是否已处理看门狗请求。然而，由于 TCP 连接确保无错误传输，所以不需要这种确认。
请注意：
由于传感器固件的限制，客户端应用程序不应每秒钟发送更多的线内看门狗进给请求（最大进给频率为 1 Hz）。
请注意：
使用命令set_scanoutput_config（参见第 3.3.6 节）启用看门狗（看门狗）或更改看门狗超时周期（watchdogtimeout）也隐含地向看门狗发送。

3.4 扫描数据的传输

扫描数据始终在数据包内传输。 一个完整的扫描通常使用多个扫描数据包传输（参见 1.1 节的基本设计考虑）。每个包包括一个通用头，一个扫描数据特定头和实际扫描数据。
可以定义多个分组类型以有效地输出不同组的扫描数据信息。这些分组类型通常遵循标准结构-仅在批量扫描数据方面不同。在批量扫描数据内，通常每个扫描点由包含有利数据量（距离，幅度等）的结构表示。
以下部分详细描述扫描数据分组。

3.4.1 基本包结构

每个数据包具有以下基本结构：
虽然数据包的结构通常看起来是固定的，但强烈建议客户端应用程序总是评估数据包大小和标题大小的条目，因为它们可能会由于未来的扩展而更改。
在分组头部的开始处的魔术字节被设计为用作连续数据流内的同步标记。如果不需要同步，

它可以被忽略。
通过使用报头(header_padding)中的填充字节，有效载荷数据的起始地址总是与 32 位地址边界对齐。

type	name	description
uint16	magic	magic byte (0xa25c) marking the beginning of a packet
uint16	packet_type	type of scan data packet
uint32	packet_size	overall size of this packet in bytes (header and payload)
uint16	header_size	size of header in bytes (i.e. offset to payload data)
...	...	packet type specific additional header information
uint8[]	header_padding	0-3 bytes padding (to align the header size to a 32bit boundary)
...	payload_data	packet type specific payload data

3.4.2 扫描数据头的典型结构

扫描数据包包含具有关于扫描的信息和扫描数据本身的扫描数据头。扫描数据头被设计为每个扫描数据包可以独立于属于相同扫描的其他扫描数据包进行处理。
典型的扫描数据头具有以下结构：

type	name	description
uint16	magic	magic byte (0xa25c) marking the beginning of a packet
uint16	packet_type	type of scan data packet
uint32	packet_size	overall size of this packet in bytes (header and payload)
uint16	header_size	size of header in bytes (i.e. offset to payload data)
uint16	scan_number	sequence number for scan (incremented for every scan, starting with 0, overflows)
uint16	packet_number	sequence number for packet (counting packets of a particular scan, starting with 1)
uint64	timestamp_raw	timestamp of internal clock (NTP time format)
uint64	timestamp_sync	timestamp of clock synchronized with time server (NTP time format) Note: Synchronized timestamps are currently not available and are output as zero.
uint32	status_flags	scan status flags (see section 3.4.3)
uint32	scan_frequency	frequency of head rotation (1/1000 Hz)
uint16	num_points_scan	number of scan points (samples) within complete scan (depending on configured FOV)
uint16	num_points_packet	number of scan points within this packet
uint16	first_index	index of first scan point within this packet
int32	first_angle	absolute angle of first scan point in this packet (1/10 000°)
int32	angular_increment	delta angle between two scan points (1/10 000°) (CCW rotation: positive increment, CW rotation: negative increment)
uint32	iq_input	reserved – all bits zero for devices without switching I/Q
uint32	iq_overload	reserved – all bits zero for devices without switching I/Q
uint8[]	header_padding	0-3 bytes padding (to align the header size to a 32bit boundary)
...	scandata	packet type specific scan data

请注意：
字段num_points_scan表示每个记录扫描输出的扫描点的总数。它始终等于samples_per_scan或max_num_points_scan，对于特定的扫描数据连接，其中较小者。有关此事项的更多详情，请参阅第 3.2.4 节。
请注意：
指定的角度值分辨率为1/10000°通常由于值的十进制范围而容易出现舍入误差。为了方便，它们是报头的一部分。需要精确角度值的后续计算应通过参考其索引号，配置的角增量和配置的扫描起始角，计算每个扫描点的精确角度：

逆时针旋转：$\text{exact\_angle}_{scanpoint} = start\_angle_{scan} + index_{scanpoint} * \frac{360^o}{num\_poimt\_scan}$

顺时针选装：$\text{exact\_angle}_{scanpoint} = start\_angle_{scan} - index_{scanpoint} * \frac{360^o}{num\_poimt\_scan}$

3.4.3 扫描数据头状态标志

扫描数据头状态标志与系统状态标志类似（见第 2.8.2 节），但提供特定于扫描数据包的扫描数据的状态信息。每个扫描数据头包含一个 uint32 条目status_flags（见第 3.4.2 节），包含以下标志：

bit	flag name	description
Generic		
0	scan_data_info	Accumulative flag – set if any of the following generic flags is set
1	new_settings	System settings for scan data acquisition changed during recording of this packet. Consistency of scan data is not guaranteed (see section 3.3.6)!
2	invalid_data	Consistency of scan data is not guaranteed for this packet.
3	unstable_rotation	Scan frequency did not match set value while recording this scan data packet.
4	skipped_packets	Preceding scan data packets have been skipped due to connection issues. This might occur of the client does not receive data fast enough.
Warnings		
8	device_warning	Accumulative flag – set if device displays any warning
10	low_temperature_warning	Current device-internal temperature below warning threshold (0 °C)
11	high_temperature_warning	Current device-internal temperature above warning threshold (80 °C)
12	device_overload	Overload warning – sensor CPU overload is imminent
Errors		
16	device_error	Accumulative flag – set if device displays any error
18	low_temperature_error	Current device-internal temperature below error threshold (−10 °C)
19	high_temperature_error	Current device-internal temperature above error threshold (85 °C)
20	device_overload	Overload error – sensor CPU is in overload state
Defects		
30	device_defect	Accumulative flag – set if device detected an unrecoverable defect

请注意：

上表中未列出的所有标志均被保留，应被忽略。

3.4.4 扫描数据包类型 A-仅距离

类型 A 的扫描数据包具有以下结构：

type	name	description
packet header		
uint16	magic	magic byte (0xa25c) marking the beginning of a packet
uint16	packet_type	type of scan data packet: 0x0041 (ASCII character 'A')
uint32	packet_size	overall size of this packet in bytes (header and payload)
uint16	header_size	size of header in bytes (i.e. offset to payload data)
uint16	scan_number	sequence number for scan (incremented for every scan, starting with 0, overflows)
uint16	packet_number	sequence number for packet (counting packets of a particular scan, starting with 1)
uint64	timestamp_raw	timestamp of internal clock (NTP time format)
uint64	timestamp_sync	timestamp of clock synchronized with time server (NTP time format) Note: Synchronized timestamps are currently not available and are output as zero.
uint32	status_flags	scan status flags (see section 3.4.3)
uint32	scan_frequency	frequency of head rotation (1/1000 Hz)
uint16	num_points_scan	number of scan points (samples) within complete scan (depending on configured FOV)
uint16	num_points_packet	number of scan points within this packet
uint16	first_index	index of first scan point within this packet
int32	first_angle	absolute angle of first scan point in this packet (1/10 000°)
int32	angular_increment	delta angle between two scan points (1/10 000°) (CCW rotation: positive increment, CW rotation: negative increment)
uint32	iq_input	reserved – all bits zero for devices without switching I/Q
uint32	iq_overload	reserved – all bits zero for devices without switching I/Q
uint8[]	header_padding	0-3 bytes padding (to align the header size to a 32bit boundary)
scan point data		
uint32	distance	measured distance (in mm) Invalid measurements return 0xFFFFFFFF.

请注意：

字段num_points_scan表示每个记录扫描输出的扫描点的总数。它始终等于samples_per_scan或max_num_points_scan，对于特定的扫描数据连接，其中较小者。有关此事项的更多详情，请参阅第 3.2.4 节。

3.4.5 扫描数据包类型 B-距离和幅度

类型 B 的扫描数据包具有以下结构：

type	name	description
packet header		
uint16	magic	magic byte (0xa25c) marking the beginning of a packet
uint16	packet_type	type of scan data packet: 0x0042 (ASCII character 'B')
uint32	packet_size	overall size of this packet in bytes (header and payload)
uint16	header_size	size of header in bytes (i.e. offset to payload data)
uint16	scan_number	sequence number for scan (incremented for every scan, starting with 0, overflows)
uint16	packet_number	sequence number for packet (counting packets of a particular scan, starting with 1)
uint64	timestamp_raw	timestamp of internal clock (NTP time format)
uint64	timestamp_sync	timestamp of clock synchronized with time server (NTP time format) Note: Synchronized timestamps are currently not available and are output as zero.
uint32	status_flags	scan status flags (see section 3.4.3)
uint32	scan_frequency	frequency of head rotation (1/1000 Hz)
uint16	num_points_scan	number of scan points (samples) within complete scan (depending on configured FOV)
uint16	num_points_packet	number of scan points within this packet
uint16	first_index	index of first scan point within this packet
int32	first_angle	absolute angle of first scan point in this packet (1/10 000°)
int32	angular_increment	delta angle between two scan points (1/10 000°) (CCW rotation: positive increment, CW rotation: negative increment)
uint32	iq_input	reserved – all bits zero for devices without switching I/Q
uint32	iq_overload	reserved – all bits zero for devices without switching I/Q
uint8[]	header_padding	0-3 bytes padding (to align the header size to a 32bit boundary)
scan point data		
uint32	distance	measured distance (in mm) Invalid measurements return 0xFFFFFFFF.
uint16	amplitude	measured amplitude (padded 12bit value – most significant bits are zero) Please see section 3.1.4 for a description of amplitude data values.

请注意：

字段num_points_scan表示每个记录扫描输出的扫描点的总数。它始终等于samples_per_scan或max_num_points_scan，对于特定的扫描数据连接，其中较小者。有关此事项的更多详情，请参阅第 3.2.4 节。

3.4.6 扫描数据包类型 C - 距离和幅度（紧凑）

类型 C 的扫描数据包具有以下结构：

type	name	description
packet header		
uint16	magic	magic byte (0xa25c) marking the beginning of a packet
uint16	packet_type	type of scan data packet: 0x0043 (ASCII character 'C')
uint32	packet_size	overall size of this packet in bytes (header and payload)
uint16	header_size	size of header in bytes (i.e. offset to payload data)
uint16	scan_number	sequence number for scan (incremented for every scan, starting with 0, overflows)
uint16	packet_number	sequence number for packet (counting packets of a particular scan, starting with 1)
uint64	timestamp_raw	timestamp of internal clock (NTP time format)
uint64	timestamp_sync	timestamp of clock synchronized with time server (NTP time format) Note: Synchronized timestamps are currently not available and are output as zero.
uint32	status_flags	scan status flags (see section 3.4.3)
uint32	scan_frequency	frequency of head rotation (1/1000 Hz)
uint16	num_points_scan	number of scan points (samples) within complete scan (depending on configured FOV)
uint16	num_points_packet	number of scan points within this packet
uint16	first_index	index of first scan point within this packet
int32	first_angle	absolute angle of first scan point in this packet (1/10 000°)
int32	angular_increment	delta angle between two scan points (1/10 000°) (CCW rotation: positive increment, CW rotation: negative increment)
uint32	iq_input	reserved – all bits zero for devices without switching I/Q
uint32	iq_overload	reserved – all bits zero for devices without switching I/Q
uint8[]	header_padding	0-3 bytes padding (to align the header size to a 32bit boundary)
scan point data		
uint20	distance	measured distance (in mm) – maximum representable value is 1 km Invalid measurements return 0xFFFFF.
uint12	amplitude	measured amplitude Please see section 3.1.4 for a description of amplitude data values.

类型 C 的扫描数据包与类型 B 的不同之处仅在于值的距离和幅度的二进制大小。对于类型

C，这些值被编码为 uint32 类型中的位字段。
请注意：
字段num_points_scan表示每个记录扫描输出的扫描点的总数。它始终等于samples_per_scan或max_num_points_scan，对于特定的扫描数据连接，其中较小者。有关此事项的更多详情，请参阅第 3.2.4 节。

3.5 使用 TCP 的数据传输

当使用基于 TCP 的通道进行扫描数据传输时，扫描数据将被分成一个或多个扫描数据包。新的扫描将始终以新的扫描数据包开始。
请注意：
当前扫描数据根据以太网帧的大小（通常为 1500 字节）被划分为几个扫描数据分组。
请注意：
只有在客户端成功接收到最后一次扫描后，才会传输新的扫描。如果扫描数据的接收由于传输错误或慢客户端响应而延迟，则传感器可能跳过单个扫描分组的传输或完全扫描。

3.6 使用 UDP 的数据传输

当使用基于 UDP 的信道进行扫描数据传输时，扫描数据将根据以太网帧的大小（通常为 1500 字节）划分为几个扫描数据包。新的扫描将始终以新的扫描数据包开始。
如果在传输期间一些扫描数据分组丢失，则不提供错误恢复。客户端应用可以使用成功接收的扫描数据分组(包含部分扫描数据)，因为每个扫描数据分组包括全扫描数据头部并且可以被单独处理。
请注意：
传感器使用特殊的实时(RT)任务为 UDP 扫描数据输出，以最小化延迟。此 RT 任务当前仅可用于单个 UDP 客户端连接。附加（并行）UDP 客户端连接由非 RT 任务处理，并且可能显示较差的时间行为。

4 使用 HMI LED 显示屏

本章将详细描述 R2000 的 HMI LED 显示以及如何使用它来显示特定应用信息。

4.1 技术概述

R2000 设备系列具有多功能 HMI LED 显示屏。该显示器由安装在旋转传感器头的一个边缘上的 24 个 LED 的阵列产生，利用人眼的所谓的视觉持续性（POV）特性。通过在传感器头的旋转期间快速更新 LED 阵列，创建具有 24 行和 252 列的虚拟光栅图形。为了获得最佳可读性，建议使用高于 35 Hz 的扫描频率（参见第 2.6 节中的scan_frequency参数）。

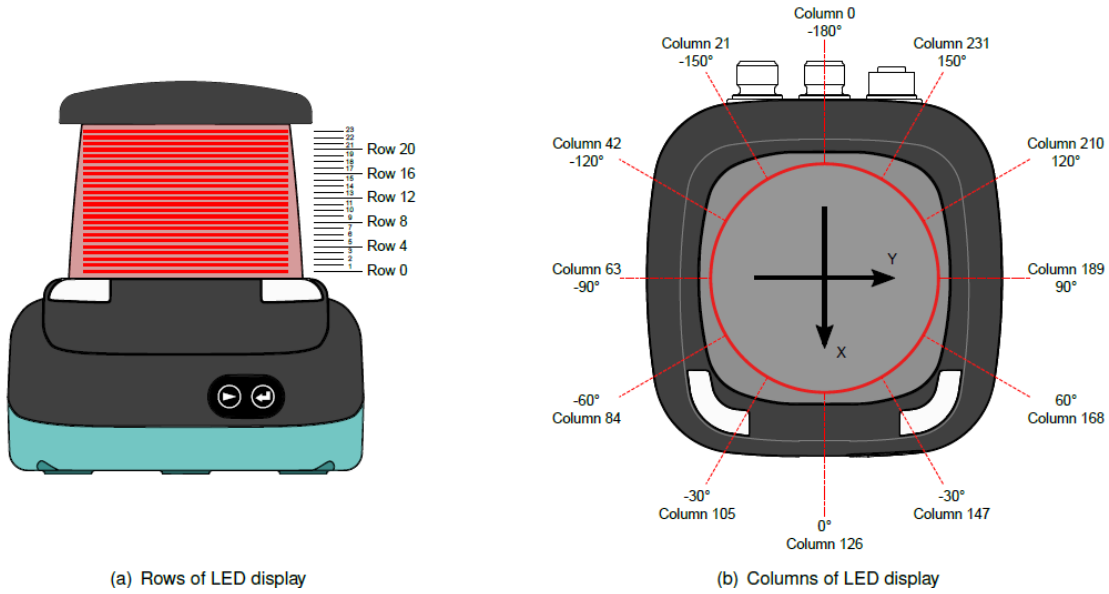

图 4.1：HMI 显示坐标系

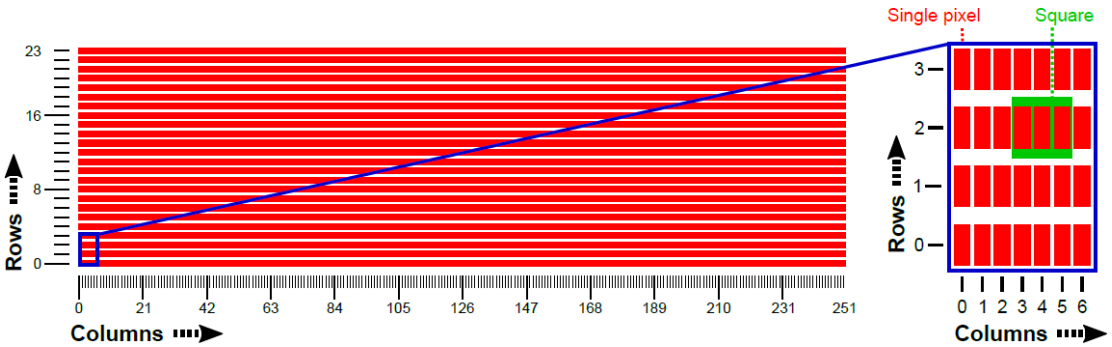

图 4.2：HMI 显示像素布局(252 x 24 像素)

图 4.1 显示了 LED 显示器相对于传感器坐标系(如 3.1.1 节所定义)的位置。LED 显示屏上的信息通常以 0° 为中心在传感器的前面(列 126)显示。显示区域从-180° 传感器背面的第 0 列开始。列以大约在 178.6° 地方的数学正序排列直到第 251 列。在传感器上从列 251 到列 0 的转换是无缝的，使得可用的视场为 360°。显示内容的呈现独立于传感器的旋转方向（参见 2.6 节中的参数scan_direction）-传感器固件负责所有必要的调整。

图 4.2 显示了像素布局（2D 位图）的二维表示。这是一个简化的视图，因为曲率和 360° 未

显示出真实的环绕性质。物理显示区域的高度约为 48mm，宽度为 170mm。这导致约 38dpi 的水平像素密度（分辨率）和约 12dpi 的垂直像素密度。三倍高的水平分辨率意味着需要组合三个水平像素以便在 HMI LED 显示器上显示单个正方形“像素”。
应用开发者可以利用 LED 显示器来显示自定义文本消息或自定义位图图像。以下部分详细描述了这些用例。

4.2 显示自定义文本消息

使用短文本消息是在R2000 HMI LED显示屏上显示自定义信息的最简单方法。传感器支持两种不同的显示文本模式：静态文本消息的模式，即使在电源循环(断电)之后仍被保留，并且模式用于由客户端应用频繁地更新的遗失性文本消息。

4.2.1 概述

文本显示有两个独立的文本行-一个在显示器的上半部分，一个在下半部分。文本以 8 pt高度的固定宽度字体显示（使用 24 个显示行中的 10 个来显示）。每个文本行的长度最多为 30 个字符，水平居中显示在传感器的前面。显示器支持从 UTF-8 代码范围中选择典型的特殊字符（例如，变音符号）。不支持的特殊字符由问号字符（'？'）替换。

4.2.2 静态文本消息（静态文本）

显示模式static_text允许客户端应用程序在 HMI LED 显示屏上最多显示两行静态文本。应用程序文本行存储在非易失性存储器中的参数hmi_static_text_1和hmi_static_text_2中，即在功率周期期间内容不会丢失。显示模式特别适合于显示很少更新的信息，例如，传感器的标识字符串。参数hmi_static_text_1和hmi_static_text_2仅在请求时重置（例如，在加载出厂默认值时）。默认内容为"Pepperl + Fuchs"和"R2000"。
显示静态文本消息的步骤如下：
1.使用set_parameter将上部显示行的文本写入hmi_static_text_1。
2.使用set_parameter将下显示行的文本写入hmi_static_text_2。
3.使用set_parameter将参数hmi_display_mode设置为static_text，以启用静态文本显示。
请注意：
由于每次对hmi_display_mode，hmi_static_text_1和hmi_static_text_2的写访问都会触发对非易失性存储器的写访问，并且写周期数有限，因此强烈建议仅在必要时写入这些参数。
命令示例
查询选择显示模式static_text：
http://< IP address >/cmd/set_parameter?hmi_display_mode = static_text
查询将显示的文本设置为“Hello World!”：
http://< IP address >/cmd/set_parameter?hmi_static_text_1 = Hello&hmi_static_text_2 = World!

4.2.3 动态文本消息（应用程序文本）

显示模式application_text允许客户端应用程序在 HMI LED 显示屏上最多显示两行自定义文

本。应用程序文本行存储在参数hmi_application_text_1和hmi_application_text_2中，仅使用易失性存储器，即内容在复位期间丢失。因此，这种显示模式特别适合于显示频繁更新的信息，例如。处理传感器扫描数据的客户端应用程序的状态信息。默认情况下（例如，开机后）hmi_application_text_1和hmi_application_text_2为空。

显示应用程序文本消息的步骤如下：

1.使用set_parameter将参数hmi_display_mode设置为application_text，以启用应用程序文本显示。

2.使用set_parameter将上部显示行的文本写入hmi_application_text_1。

3.使用set_parameter将下显示行的文本写入hmi_application_text_2。

请注意：

由于对hmi_display_mode的每次写入访问都会触发对具有有限数量的写周期的非易失性存储器的写访问，因此强烈建议仅将其写入以选择显示模式。对于后续内容更新，仅写入hmi_application_text_1和hmi_application_text_2。

命令示例

查询选择显示模式 application_text:

http://< IP address >/cmd/set_parameter? hmi_display_mode = application_text

Query for setting the displayed text to 'My status message':

http://<IP address>/cmd/set_parameter?hmi_application_text_1=My status&hmi_application_text_2=message

4.3 显示自定义位图

或者对于简单的文本消息（如 4.2 节所述），传感器允许客户端应用程序在 HMI LED 显示屏上显示自定义位图。这给出了关于所显示内容的最大灵活性，但是需要由客户端固件进行更复杂的准备。与用于文本消息的显示模式类似，传感器提供用于即使在电源周期之后仍被保留的静态位图（标志）的模式，以及由客户端应用更频繁地更新的相当动态图形的模式。

4.3.1 概述

第 4.1 节已经涵盖了 LED 显示屏技术实现的各种细节。本节重点介绍如何将显示像素映射到二进制帧缓冲区以及如何将该数据传输到传感器。

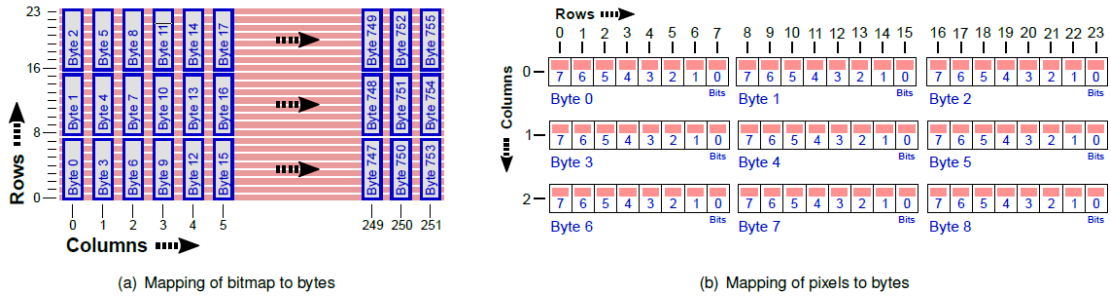

图 4.3：HMI 显示像素的字节映射

图 4.3 显示了显示像素到二进制帧缓冲区的映射。LED 显示屏的每个像素（见图 4.2）由该帧缓冲器中的单个位表示。完整显示图像的 6048 个像素（24*252 = 6048）映射到 756B(6048 /8 = 756)的帧缓冲器中。映射从 2D 显示图像的左下角的列 0 的行 0 开始：像素（0; 0）映射到字节 0 位 7，像素（0; 1）映射到字节 0 位 6，等等。第一列的最后一个像素（0; 23）映射

到字节 3 位 0.第二列的第一个像素（1; 0）映射到字节 4 位 7.该映射方案逐列继续，直到最后一个像素在 2D 显示图像的右上角中的位（251; 23）填充字节 755 的位 0。
位图访问 HMI LED 显示屏总是更新整个显示帧缓冲区。必要的数据以二进制形式存储在显示参数hmi_static_logo和hmi_application_bitmap中-有关详细信息，请参见后续部分。使用set_parameter写入这些参数需要将二进制内容编码为命令 URI 的base64url字符串（请参见第 1.2.1 和 1.2.2 节）。读取参数返回一个 base64 编码字符串作为 JSON 编码的一部分（见第 1.2.3 节）。有关二进制参数类型的更详细描述，请参阅第 2.1 节。

4.3.2 静态位图

参数hmi_static_logo允许定制 HMI 显示模式static_logo所示的图形。静态位图存储在非易失性存储器中，即内容在电源周期期间不丢失。因此，这种显示模式特别适用于显示很少更新的信息，例如。定制公司标志。参数hmi_static_logo仅在请求时重置（例如，在加载出厂默认值时）。默认值显示Pepperl + Fuchs徽标。
自定义静态徽标的步骤如下：
1.使用set_parameter将自定义位图写入hmi_static_logo。
2.使用set_parameter将参数hmi_display_mode设置为static_logo，以显示位图。
请注意：
由于对hmi_display_mode和hmi_static_logo的每次写入访问都会触发对具有有限数量写周期的非易失性存储器的写访问，因此强烈建议仅在必要时写入这些参数。

4.3.3 应用程序位图

显示模式application_bitmap使客户端应用程序能够在 HMI LED 显示屏上显示自定义位图图像。位图仅使用易失性存储器存储在参数hmi_application_bitmap中，即内容在复位期间丢失。因此，这种显示模式特别适合于显示频繁更新的信息，例如。状态图形的客户端应用程序处理传感器扫描数据。默认情况下（例如，开机后）hmi_application_bitmap为空。
显示应用程序位图的步骤如下：
1.通过使用set_parameter将参数hmi_display_mode设置为static_logo，将显示模式切换为显示位图。
2.使用set_parameter将应用程序位图写入参数hmi_application_bitmap。
请注意：
由于对hmi_display_mode的每次写入访问都会触发对具有有限数量的写周期的非易失性存储器的写访问，因此强烈建议仅将其写入以选择显示模式。对于后续内容更新，仅写入hmi_application_bitmap。
请注意：
HMI LED 显示屏的接口当前未设计用于实时更新。建议更新应用程序位图的频率不要超过每秒一次（更新速率为 1 Hz）。在更快的更新的情况下的行为
显示未定义。

4.3.4 转换 HMI 显示的图形

要将现有图形转换为 HMI LED 显示屏，建议执行以下步骤：
1.在水平方向上用因子 3 拉伸图形，以补偿不对称的显示分辨率。
2.将图像修剪为 2：21 的纵横比（垂直：水平）。请记住，图像显示在 360 度表面，因此为不同的视角重复图像可能是一个好主意。一个尝试和值得信赖的方法是修剪将图像转换为 2：7 的纵横比，然后在水平方向上重复该图像三次。
3.将图像缩小到 24 x 252 像素的分辨率。
4.将图像缩小为仅有两种颜色的黑白图形。
5.如有必要，通过从转换过程中移除伪像来手动优化生成的低分辨率图形。
6.将图像保存为支持单色图像（每像素 2 位）的通用图形文件格式。
7.使用常用的图像转换工具（如Image Magick）将图像文件转换为二进制格式。以下命令使用Image Magick转换的工具将位图转换为正确的原始二进制数据：
> convert input.bmp －rotate 90 －negate GRAY：output.bin
生成的二进制文件（上例中的′output.bin′）的大小应该恰好为 756B。
8.最后，将二进制数据存储在hmi_static_logo或hmi_application_bitmap中。
数据需要编码为 base64url string [8]，长度为 1008B。强烈建议使用矢量图形作为创建 HMI LED 显示屏位图的源。这样，可以避免许多转换伪像，导致更高的图像质量。