TeraCota Fieldbus Interface Specification

Overview

The purpose of the fieldbus interface is to allow an external device, such as a factory PLC, to control the TeraCota instrument and to access data from each measurement. To that end, the communication that passes through the fieldbus interface is bidirectional. Communication from the factory (Master) to the TeraCota instrument (Slave) will be referred to as MOSI (Master Out, Slave In). Conversely, communication from the TeraCota instrument to the factory will be referred to as MISO (Master In, Slave Out).

Results from each measurement will be placed into a First In, First Out (FIFO) buffer. These results can be requested at asynchronously through the fieldbus gateway.

Within this document the TeraCota control state machine is presented. This is followed by the MOSI data structures and definition of each field. Then the MISO data structure is outlined and each field is defined. The document concludes with an overview of the control flow for a factory -> TeraCota interaction.

Note that this document is applicable for both single and dual-instrument systems. In the case of the single instrument system the fields for the second instrument can simply be ignored.

(Note, the term “sensor” and “instrument” are used somewhat synonymously)

TeraCota Control State Machine

The state machine for the TeraCota instrument is shown below (Figure 1). Opening the TeraCota application will automatically initialise the system and put it into the READY state.

Typical transition times, and suggested timeouts for the control programme are listed below:

- REINITIALISE transition, usually completed in less than 60s, timeout 120s
- START/STOP SCANNING transition, usually completed in 10s, timeout 30s
- DO MEASUREMENT transition, usually completed in less than 1s, timeout 10s and will automatically return to SCANNING after 5s for Point measurement and 12s for AutoAlignPoint measurement.
- GOTO STANDBY transition, usually completed in 10s, timeout 30s

The automatic transition from ‘Measuring’ to ‘Ready’ can occur if there is a hardware error during a measurement. In this case the best course of action is to restart the scanner and attempt the measurement again.

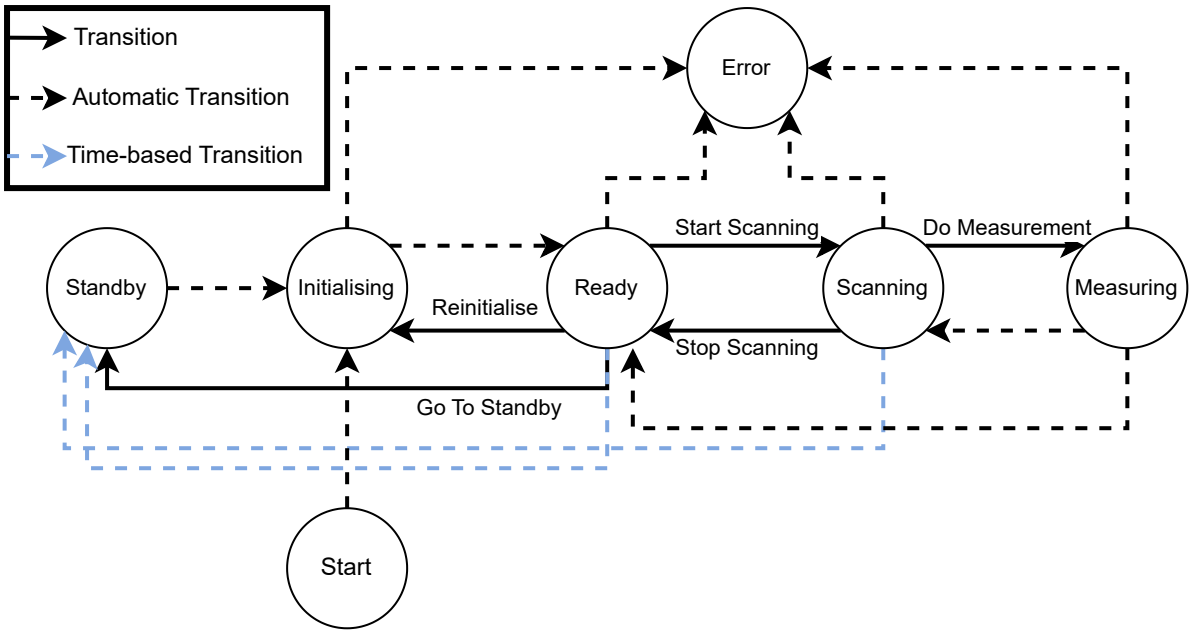


Figure 1: TeraCota Control State Machine

Fieldbus Configuration

Both the MOSI and MISO data structures feature several ‘core’ objects which are used to by TeraCota to perform essential functions. An example core object would be the the Job ID which is used to uniquely define a set of measurements; changes to this object will initiate file export (see Summary of Fieldbus Interface Control Flow section). However, TeraCota is sufficiently flexible that additional ‘meta’ objects can be defined that will not affect the operation of TeraCota but instead can be exported with other core measurement data. The configuration is defined by a JSON file that is specified in the YAML configuration file. The data structure for both the MOSI and MISO are defined by this single file.

Configuration File

The configuration is defined through a JSON file. An example which outlines the core/default data structures is given below.

Both the MISO and MOSI data structures are limited to a size of 200 bytes for both instruments. By default it is assumed that the data structure below is little-endian, but this can switched in the TeraCota configuration file.

Example Core Configuration File

```
{
  "mosi": {
    "_teracota_control": ["B","TeraCota Control"],
    "_measurement_type": ["B","Measurement Type"],
    "_fieldbus_job_id": ["H","Job ID"],
    "_vehicle_id": ["I","Vehicle ID"],
    "location_id": ["H","Location ID"],
    "body_id": ["H","Body ID"],
    "_paint_code_1": ["B","Paint Code Char1"],
    "_paint_code_2": ["B","Paint Code Char2"],
    "_paint_code_3": ["B","Paint Code Char3"],
```

```
    "_result_access": ["B","Result Access"],
    "spacer1": ["61B","Spacing before second instrument"]
  },
  "miso": {
    "teracota_status": ["B","TeraCota Stasus"],
    "teracota_heartbeat": ["B","TeraCota Heartbeat"],
    "teracota_error_code": ["B","TeraCota Error Code"],
    "teracota_heatsink_tempC": ["B","TeraCota Heatsink TempC"],
    "result_buffer_size": ["B","Results Buffer Size"],
    "result_buffer_pending": ["B","Results Buffer Pending"],
    "result_job_id": ["H","Results Job ID"],
    "result_vehicle_id": ["I","Results Vehicle ID"],
    "result_location_id": ["H","Results Location ID"],
    "result_body_id": ["H","Results Body ID"],
    "result_status": ["H","Results Status"],
    "result_layer_1_thickness": ["H","Results Layer 1 Thickness"],
    "result_layer_2_thickness": ["H","Results Layer 2 Thickness"],
    "result_layer_3_thickness": ["H","Results Layer 3 Thickness"],
    "result_layer_4_thickness": ["H","Results Layer 4 Thickness"],
    "result_layer_5_thickness": ["H","Results Layer 5 Thickness"],
    "result_layer_6_thickness": ["H","Results Layer 6 Thickness"],
    "result_layer_1_uncertainty": ["H","Results Layer 1 Uncertainty"],
    "result_layer_2_uncertainty": ["H","Results Layer 2 Uncertainty"],
    "result_layer_3_uncertainty": ["H","Results Layer 3 Uncertainty"],
    "result_layer_4_uncertainty": ["H","Results Layer 4 Uncertainty"],
    "result_layer_5_uncertainty": ["H","Results Layer 5 Uncertainty"],
    "result_layer_6_uncertainty": ["H","Results Layer 6 Uncertainty"],
    "result_layer_1_status": ["B","Results Layer 1 Status"],
    "result_layer_2_status": ["B","Results Layer 2 Status"],
    "result_layer_3_status": ["B","Results Layer 3 Status"],
    "result_layer_4_status": ["B","Results Layer 4 Status"],
    "result_layer_5_status": ["B","Results Layer 5 Status"],
    "result_layer_6_status": ["B","Results Layer 6 Status"],
    "result_alignment_status": ["H","Results HAS Alignment"],
    "result_has_axis_1": ["h","Results HAS Axis 1"],
    "result_has_axis_2": ["h","Results HAS Axis 2"],
    "result_has_axis_3": ["h","Results HAS Axis 3"],
    "result_transform_status": ["B","Results Transform Status"],
    "result_transformed_axis_1": ["h","Results Transformed Axis 1"],
    "result_transformed_axis_2": ["h","Results Transformed Axis 2"],
    "result_transformed_axis_3": ["h","Results Transformed Axis 4"],
    "result_transformed_axis_4": ["h","Results Transformed Axis 4"],
    "result_transformed_axis_5": ["h","Results Transformed Axis 5"],
    "result_transformed_axis_6": ["h","Results Transformed Axis 6"],
    "spacer1": ["22B","Spacing before second instrument"]
  },
  "formatters": {
    "measurement_name_format": [
      "vehicle_id",
      "job_id",
      "location_id",
          "body_id"
    ],
    "filename_format": [
      "job_id"
    ],
    "vehicle_id_format": [
      "_vehicle_id"
    ]
  },
  "flags": {
    "_use_fieldbus_job_id": true
  },
  "measurement_meta_data_map": {
          "job_id": "job_id",
          "vehicle_id": "vehicle_id",
          "body_id": "body_id",
```

```
            "location_id": "location_id"
    }

}
```

Defining data structures

The first two fields are named mosi and miso and these define the data structure for the MOSI and MISO repsectively. Within this field a dictionary is used where the key is the variable name used within TeraCota and the value is a list of two configurable variables. The first item in this list is ia format string the defines the object type. This follows the convetion of the module struct. The second item in the list is a comma delimited comment field.This is used to provided context for the variable ad is especially useful when defining *meta* objects.

A variable named *spacerN* , where N is an integer, can be used to add gaps in the data structure where TeraCota will ignore what is present. The naming allows multiple spacers to be added. The most common use of a spacer is to add a space between mutliple sensors.

Note that some variables are preceded by an underscore e.g. _teracota_control in order to differentiate them from other variable names.

Configuring the Job ID

The Job ID is used to define a unique collection of measurements. This can either be defined through the fieldbus, or internally in TeraCota. If the Job ID is be defined by the fieldbus then the flag _use_fieldbus_job_id must be set to true. Alternatively, if the Job ID is to be tracked by TeraCota then this flag should be set to false. Regardless of whether this flag is true or false the MOSI data strucutre must still contain an item named _fieldbus_job_id

Configuring the Vehicle ID

The vehicle ID can be constructed in multiple ways. In some applications the Vehicle ID consists only of integers e.g. 12345678. In other cases there may be a character used to define it e.g. A12345678. In order to define the format of this Vehicle ID a configurable formatter exists ‘vehicle_id_format’. The items in the list defined by this field will be concatenated without any delimter to make the final vehicle ID. For example, a purely integer based Vehicle ID will be defined with [“_vehicle_id”]. Whereas a Vehicle which contains characters will be constructed as [“_shared_metachar_0”,”_vehicle_id”]. Consult the meta data definition for more information on meta data.

Configuring Measurement names

‘measurement_name_format’ : Measurement names will be constructed as underscore delimited with the the field described in this list. For example, a list of [“vehicle_id”, “job_id”,”location_id”] would produce a measurement name of “vehicle_id”_”job_id”_”location_id”

Configuring File names

‘filename_format’ : The base filename upon job export will be named according to the items in this list delimited by an underscore. This is a base filename format as other parameters such as a paint code and timestamp are added to filenames.

Measurement Meta Data Map

This maps data objects from the fieldbus into measurement meta data such that it can be exported. This is specified as a dictionary where the key is the name given to the meta data and the value is the corresponding fieldbus value. See The example configuration with Meta Data below for an illustrative example. The following values are mandatory for the measurement meta data map :

{‘job_id’, ‘location_id’, ‘vehicle_id’, ‘body_id’}

Example Configuration File with Meta Data

The example below demonstrates how to define meta data variables. There are two types of variables that can be configured : characters and integers. In addition, there are two forms to these variables : single-instrument and shared. Integers that correspond to a single-instrument are added with the field ‘meta_integers’. Integers that are shared between instruments are added with the field ‘meta_shared_integers’. Characters that correspond to a single instrument are added with the field ‘_meta_chars’. Characters that are shared between instruments are added with the field ‘_meta_sharedchars’.

These meta variables can be accessed in formatters by using the relative field name followed by an index. For example, ‘meta_sharedchars_0’ in a formatter field will access the first meta_sharedchars variable and likewise ‘meta_sharedchars_1’ the second. It is possible combine the field together through an underscore delimiter e.g. ‘meta_sharedchars_0_1’. The ‘measurement_meta_data_map’ in the example below show how to map these variables to meta data.

```
{
  "mosi": {
    "_teracota_control": ["B","TeraCota Control"],
    "_measurement_type": ["B","Measurement Type"],
    "_fieldbus_job_id": ["H","Job ID"],
    "_vehicle_id": ["I","Vehicle ID"],
    "location_id": ["H","Location ID"],
    "body_id": ["H","Body ID"],
    "_paint_code_1": ["B","Paint Code Char1"],
    "_paint_code_2": ["B","Paint Code Char2"],
    "_paint_code_3": ["B","Paint Code Char3"],
    "_result_access": ["B","Result Access"],
    "meta_integers": ["BHI","A 1 byte int, A 2 byte int, A 4 byte int"],
    "meta_sharedintegers": ["BHI","A 1 byte int, A 2 byte int, A 4 byte int"],
    "_meta_chars": ["4B","Char1,Char2,Char3,Char4"],
    "spacer1": ["1B"," Some spacing"],
    "_meta_sharedchars": ["4B","SChar1,SChar2,SChar3,SChar4"],
    "spacer2": ["61B","Spacing before second instrument"]
  },
  "miso": {
    "teracota_status": ["B","TeraCota Stasus"],
    "teracota_heartbeat": ["B","TeraCota Heartbeat"],
    "teracota_error_code": ["B","TeraCota Error Code"],
    "teracota_heatsink_tempC": ["B","TeraCota Heatsink TempC"],
    "result_buffer_size": ["B","Results Buffer Size"],
```

```
    "result_buffer_pending": ["B","Results Buffer Pending"],
    "result_job_id": ["H","Results Job ID"],
    "result_vehicle_id": ["I","Results Vehicle ID"],
    "result_location_id": ["H","Results Location ID"],
    "result_body_id": ["H","Results Body ID"],
    "result_status": ["H","Results Status"],
    "result_layer_1_thickness": ["H","Results Layer 1 Thickness"],
    "result_layer_2_thickness": ["H","Results Layer 2 Thickness"],
    "result_layer_3_thickness": ["H","Results Layer 3 Thickness"],
    "result_layer_4_thickness": ["H","Results Layer 4 Thickness"],
    "result_layer_5_thickness": ["H","Results Layer 5 Thickness"],
    "result_layer_6_thickness": ["H","Results Layer 6 Thickness"],
    "result_layer_1_uncertainty": ["H","Results Layer 1 Uncertainty"],
    "result_layer_2_uncertainty": ["H","Results Layer 2 Uncertainty"],
    "result_layer_3_uncertainty": ["H","Results Layer 3 Uncertainty"],
    "result_layer_4_uncertainty": ["H","Results Layer 4 Uncertainty"],
    "result_layer_5_uncertainty": ["H","Results Layer 5 Uncertainty"],
    "result_layer_6_uncertainty": ["H","Results Layer 6 Uncertainty"],
    "result_layer_1_status": ["B","Results Layer 1 Status"],
    "result_layer_2_status": ["B","Results Layer 2 Status"],
    "result_layer_3_status": ["B","Results Layer 3 Status"],
    "result_layer_4_status": ["B","Results Layer 4 Status"],
    "result_layer_5_status": ["B","Results Layer 5 Status"],
    "result_layer_6_status": ["B","Results Layer 6 Status"],
    "result_alignment_status": ["H","Results HAS Alignment"],
    "result_has_axis_1": ["h","Results HAS Axis 1"],
    "result_has_axis_2": ["h","Results HAS Axis 2"],
    "result_has_axis_3": ["h","Results HAS Axis 3"],
    "result_transform_status": ["B","Results Transform Status"],
    "result_transformed_axis_1": ["h","Results Transformed Axis 1"],
    "result_transformed_axis_2": ["h","Results Transformed Axis 2"],
    "result_transformed_axis_3": ["h","Results Transformed Axis 4"],
    "result_transformed_axis_4": ["h","Results Transformed Axis 4"],
    "result_transformed_axis_5": ["h","Results Transformed Axis 5"],
    "result_transformed_axis_6": ["h","Results Transformed Axis 6"],
    "spacer1": ["22B","Spacing before second instrument"]
  },
  "formatters": {
    "measurement_name_format": [
      "vehicle_id",
      "job_id",
      "location_id",
          "body_id"
    ],
    "filename_format": [
      "job_id"
    ],
    "vehicle_id_format": [
      "_vehicle_id","meta_sharedchars_2"
    ]
  },
  "flags": {
    "_use_fieldbus_job_id": true
  },

  "measurement_meta_data_map": {
          "job_id": "job_id",
          "vehicle_id": "vehicle_id",
          "body_id": "body_id",
          "location_id": "location_id",
          "StyleNumber":"meta_chars_0_1"
  }

}
```

Green laser control

It is possible to control the green alignment laser through the fieldbus. The purpose of the green alignment laser is to give a visual of where the sensor is pointing. Before the laser can be used the TeraCota application must be placed in Teach mode through the GUI. This is for laser safety reasons, for more information consult the full TeraCota user guide. Moreover, the green laser can only be used if TeraCota is in the 'SCANNING' State. When in Teach mode the green laser will automatically start and stop with the TeraCota scanner.

In order to control the green laser the user may use the '_green_laser_ctrl' variable (see the in MOSI section below). A user may only control the laser if the MISO bit 'green_laser_access' has a value of 1 (see the MISO section for more information). The status of the laser is given by the 'green_laser_status' bit (see MISO section).

Core MOSI Data Structure

The data structure shown within this section is that defined by the default configuration file shown in the section above.

The data structure is split into two identical data blocks (one for each sensor). There are several 'shared' objects which will be ignored in Sensor#2. In the case of these objects the values given for Sensor#1 will be applied to both sensors e.g. Job ID. If working with a single sensor then only values for Sensor#1 need to be modified.

MOSI data must be held for a minimum of 200ms as TeraCota uses a 100ms poll rate. For example, when the control program sets data before measurement (e.g. Location ID etc.), allow 200ms for data to be read before sending the command to start measurement. Multiple write operations can occur at the same time. For example, updating the Location ID and Paint Code.

The total size of the MOSI data structure will be 200 bytes but not all of the bytes are monitored by the TeraCota application. It is recommended that the PLC interface code writes 200 bytes to negate any potential network issues and to be robust to any future additions.

Core MOSI Data Structure

Byte Index	Python Variable Name	Size	Comment
0	_teracota_control	1	TeraCota Control
1	_measurement_type	1	Measurement Type
2	_fieldbus_job_id	2	Job ID
4	_vehicle_id	4	Vehicle ID
8	location_id	2	Location ID
10	body_id	2	Body ID
12	_paint_code_1	1	Paint Code Char1
13	_paint_code_2	1	Paint Code Char2
14	_paint_code_3	1	Paint Code Char3
15	_result_access	1	Result Access
16	spacer1	61	Spacing before second instrument
77	_teracota_control	1	TeraCota Control

Byte Index	Python Variable Name	Size	Comment
78	_measurement_type	1	Measurement Type
79	_fieldbus_job_id	2	Job ID
81	_vehicle_id	4	Vehicle ID
85	location_id	2	Location ID
87	body_id	2	Body ID
89	_paint_code_1	1	Paint Code Char1
90	_paint_code_2	1	Paint Code Char2
91	_paint_code_3	1	Paint Code Char3
92	_result_access	1	Result Access
93	spacer1	61	Spacing before second instrument

Mandatory Fields

The following variables are mandatory in the MOSI field :

{‘_fieldbus_job_id’, ‘_paint_code_1’, ‘location_id’, ‘_teracota_control’, ‘_vehicle_id’, ‘_measurement_type’, ‘_paint_code_3’, ‘_result_access’, ‘body_id’, ‘_paint_code_2’}

Optional Fields

The following variables are optional in the MOSI field : {‘meta_integers’, ‘_green_laser_ctrl’, ‘_meta_chars’, ‘meta_sharedintegers’, ‘_meta_sharedchars’}

Write offsets

Empirically, we have found that some combination of hardware and firmware bugs can cause an offset in the above byte index when data is transmitted. An offset has not been observed for the current setup but we recommend that the PLC interfacing code could handle the case where this offset is non-zero.

TeraCota Control (_teracota_control)

Type=uint8

This field controls the TeraCota control commands that are shown in the state machine (Figure 1). The Lo-Byte is the TeraCota control command and the Hi-Byte is reserved.

Commands are applied in an edge-derived manner i.e. when the command value changes. If it is necessary to repeat a command, the controller should send a Command 0 (No-Op) update between command values to ensure the repeated command is applied.

For further context of these commands see the example flow control in Figure 3.

CommandValue	Description
0	No-Op.
1	Start Scanning

CommandValue	Description
2	Stop Scanning
3	Goto Standby
4	Reinitialise
5	Do Measurement

Measurement Type (_measurement_type)

Type=uint8

This selects the type of measurement for TeraCota to perform. The Lo-Byte is the command value and theHi-Byte is reserved. AutoAlignPoint uses the TeraCota Head Alignment System (HAS) to optimise THz alignment/focus before acquiring data. After AutoAlignPoint measurement is complete the HAS is restored to the orign. Point records data directly without altering the HAS positiont. AutoAlign uses the TeraCota HAS in the same was as AutoAlignPoint but it does not acquire any data, instead the HAS remains in the optimal position. ResetAlign will move the HAS back to the origin. TeraCota should be in the SCANNING state when this value is updated (see Figure 3)

CommandValue	Description
0	Point
1	AutoAlignPoint
2	AutoAlign
3	ResetAlign

Result Access (_result_access)

Type=uint8

This is used to access results that are stored in the results buffer.

Commands are applied in an edge-derived manner i.e. when the command value changes. If it is necessary to repeat a command, the controller should send a Command 0 (No-Op) update between command values to ensure the repeated command is applied. The Lo-Byte is the command value and the Hi-Byte is reserved.

Result Access = 1 (Pop Thickness Buffer) will trigger TeraCota to update the MISO data structure with a new result. After setting Result Access to 1, wait 100ms, then start polling Result Status. TeraCota will set Result Status = 7 to indicate the control program should wait before collecting the result. When the result is ready Result Status will take a value <7. The control program should timeout after 2s.

CommandValue	Description
0	No Op.
1	Pop Thickness Buffer
2	Clear Thickness Buffer

The result access process is illustrated below.

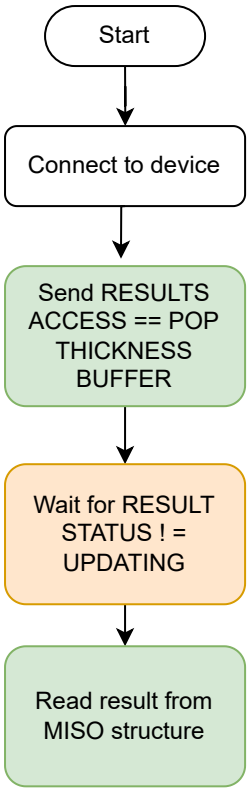


Figure 2: Results Access

If RESULT STATUS == NO RESULT, after sending the POP RESULT BUFFER command, the buffer is empty.

Optionally, read RESULT BUFFER SIZE to know when results are available.

Results can appear out of order. Check the Result Job ID and results' Location ID bytes to correctly assign values.

Result Buffer is populated asynchronously, and Result Access can be requested at any time.

Job ID (_fieldbus_job_id)

Type=uint16

A job is a collection of measurements form the TeraCota instrument. The Job ID is an integer that uniquely identifies the job. Together the Job ID, Vehicle ID, Location ID and Body ID create a unique key for the result. The Job ID should update when a new collection of measurements wants to begin.

TeraCota should be the READY state when this value is updated (see Figures 3 and 4).

Note that this always has to be specified in the data structure but is not used if the flag "_use_fieldbus_job_id" is false.

Vehicle ID (_vehicle_id)

Type=uint32

The Vehicle ID is an integer that uniquely identifies the vehicle that is being measured. Together the Job ID, Vehicle ID, Location ID and Body ID create a unique key for the result. The Vehicle ID should update when a new car enters the cell.

TeraCota should be the READY state when this value is updated (see Figures 3 and 4).

Note that the exact format of the Vehicle ID is configurable. See section of the vehicle_format_id

Location ID (location_id)

Type=uint16

The Location ID is an integer that uniquely identifies the location of the next. Together the Job ID, Vehicle ID, Location ID and Body ID create a unique key for the result. The Location ID should be updated each time TeraCota instrument is directed at a new location.

TeraCota should be the SCANNING state when this value is updated (see Figures 3 and 4).

Body ID

Type=uint16

The Body ID is an integer that uniquely identifies the body that is being measured. Together the Job ID, Vehicle ID, Location ID and Body ID create a unique key for the result.

TeraCota should be the SCANNING state when this value is updated (see Figures 3 and 4).

Paint Code First Character (_paint_code_1)

Type=uint8

This is used to instruct TeraCota on which colour paint it being measured such that the correspond calibration can be selected in TeraCota. The Lo-Byte is the value and the Hi-Byte is reserved. This can take any ASCII integer of value 33-126 (inclusive).

TeraCota should be in the SCANNING state when this value is updated (see Figure 3)

Paint Code Second Character (_paint_code_2)

Type=uint8

See Paint code first character.

Paint Code Third Character (_paint_code_3)

Type=uint8

See Paint code first character.

Green Laser Control (_green_laser_ctrl)

Type=uint8

Control the green alignment laser, note that these commands can only be actioned by TeraCota is ‘green_laser_access’ is equal to 1. This occurs when the GUI has been set in Teach mode and the TeraCota scanner is running.

CommandValue	Description
0	No Op.
1	Switch Off
2	Switch On

MISO Data Structure

The MISO (Master In, Slave Out) data will include these fields for both instruments and the values will be correct for both. There will be separate First In First Out (FIFO) buffers for the results data which can be read back in parallel.

As with the MOSI data structure the order and data type of these elements is configurable and the default order is shown below.

Core MISO Data Structure

Byte Index	Python Variable Name	Size	Comment
0	teracota_status	1	TeraCota Stasus
1	teracota_heartbeat	1	TeraCota Heartbeat
2	teracota_error_code	1	TeraCota Error Code
3	teracota_heatsink_tempC	1	TeraCota Heatsink TempC
4	result_buffer_size	1	Results Buffer Size
5	result_buffer_pending	1	Results Buffer Pending
6	result_job_id	2	Results Job ID
8	result_vehicle_id	4	Results Vehicle ID
12	result_location_id	2	Results Location ID
14	result_body_id	2	Results Body ID
16	result_status	2	Results Status
18	result_layer_1_thickness	2	Results Layer 1 Thickness
20	result_layer_2_thickness	2	Results Layer 2 Thickness
22	result_layer_3_thickness	2	Results Layer 3 Thickness
24	result_layer_4_thickness	2	Results Layer 4 Thickness
26	result_layer_5_thickness	2	Results Layer 5 Thickness
28	result_layer_6_thickness	2	Results Layer 6 Thickness
30	result_layer_1_uncertainty	2	Results Layer 1 Uncertainty
32	result_layer_2_uncertainty	2	Results Layer 2 Uncertainty
34	result_layer_3_uncertainty	2	Results Layer 3 Uncertainty
36	result_layer_4_uncertainty	2	Results Layer 4 Uncertainty

Byte Index	Python Variable Name	Size	Comment
38	result_layer_5_uncertainty	2	Results Layer 5 Uncertainty
40	result_layer_6_uncertainty	2	Results Layer 6 Uncertainty
42	result_layer_1_status	1	Results Layer 1 Status
43	result_layer_2_status	1	Results Layer 2 Status
44	result_layer_3_status	1	Results Layer 3 Status
45	result_layer_4_status	1	Results Layer 4 Status
46	result_layer_5_status	1	Results Layer 5 Status
47	result_layer_6_status	1	Results Layer 6 Status
48	result_alignment_status	2	Results HAS Alignment
50	result_has_axis_1	2	Results HAS Axis 1
52	result_has_axis_2	2	Results HAS Axis 2
54	result_has_axis_3	2	Results HAS Axis 3
56	result_transform_status	1	Results Transform Status
57	result_transformed_axis_1	2	Results Transformed Axis 1
59	result_transformed_axis_2	2	Results Transformed Axis 2
61	result_transformed_axis_3	2	Results Transformed Axis 4
63	result_transformed_axis_4	2	Results Transformed Axis 4
65	result_transformed_axis_5	2	Results Transformed Axis 5
67	result_transformed_axis_6	2	Results Transformed Axis 6
69	spacer1	22	Spacing before second instrument
91	teracota_status	1	TeraCota Stasus
92	teracota_heartbeat	1	TeraCota Heartbeat
93	teracota_error_code	1	TeraCota Error Code
94	teracota_heatsink_tempC	1	TeraCota Heatsink TempC
95	result_buffer_size	1	Results Buffer Size
96	result_buffer_pending	1	Results Buffer Pending
97	result_job_id	2	Results Job ID
99	result_vehicle_id	4	Results Vehicle ID
103	result_location_id	2	Results Location ID
105	result_body_id	2	Results Body ID
107	result_status	2	Results Status
109	result_layer_1_thickness	2	Results Layer 1 Thickness
111	result_layer_2_thickness	2	Results Layer 2 Thickness
113	result_layer_3_thickness	2	Results Layer 3 Thickness
115	result_layer_4_thickness	2	Results Layer 4 Thickness
117	result_layer_5_thickness	2	Results Layer 5 Thickness
119	result_layer_6_thickness	2	Results Layer 6 Thickness
121	result_layer_1_uncertainty	2	Results Layer 1 Uncertainty
123	result_layer_2_uncertainty	2	Results Layer 2 Uncertainty
125	result_layer_3_uncertainty	2	Results Layer 3 Uncertainty
127	result_layer_4_uncertainty	2	Results Layer 4 Uncertainty
129	result_layer_5_uncertainty	2	Results Layer 5 Uncertainty

Byte Index	Python Variable Name	Size	Comment
131	result_layer_6_uncertainty	2	Results Layer 6 Uncertainty
133	result_layer_1_status	1	Results Layer 1 Status
134	result_layer_2_status	1	Results Layer 2 Status
135	result_layer_3_status	1	Results Layer 3 Status
136	result_layer_4_status	1	Results Layer 4 Status
137	result_layer_5_status	1	Results Layer 5 Status
138	result_layer_6_status	1	Results Layer 6 Status
139	result_alignment_status	2	Results HAS Alignment
141	result_has_axis_1	2	Results HAS Axis 1
143	result_has_axis_2	2	Results HAS Axis 2
145	result_has_axis_3	2	Results HAS Axis 3
147	result_transform_status	1	Results Transform Status
148	result_transformed_axis_1	2	Results Transformed Axis 1
150	result_transformed_axis_2	2	Results Transformed Axis 2
152	result_transformed_axis_3	2	Results Transformed Axis 4
154	result_transformed_axis_4	2	Results Transformed Axis 4
156	result_transformed_axis_5	2	Results Transformed Axis 5
158	result_transformed_axis_6	2	Results Transformed Axis 6
160	spacer1	22	Spacing before second instrument

The total size of the MISO data structure will be 200 bytes but not all the bytes are written by the TeraCota application. It is recommended that the PLC interface code reads 200 bytes to negate any potential network issues and to be robust to any future additions.

Mandatory Fields

The following variables are mandatory in the MISO field :

{‘result_layer_1_status’, ‘result_layer_4_status’, ‘teracota_error_code’, ‘result_vehicle_id’, ‘result_layer_2_status’, ‘result_location_id’, ‘result_layer_6_uncertainty’, ‘result_layer_4_uncertainty’, ‘teracota_heartbeat’, ‘result_layer_3_uncertainty’, ‘result_body_id’, ‘result_layer_3_thickness’, ‘result_layer_5_status’, ‘result_layer_1_thickness’, ‘teracota_status’, ‘result_buffer_size’, ‘result_layer_5_uncertainty’, ‘result_layer_1_uncertainty’, ‘result_layer_3_status’, ‘result_layer_5_thickness’, ‘result_status’, ‘result_layer_2_uncertainty’, ‘result_layer_6_status’, ‘result_layer_2_thickness’, ‘result_job_id’, ‘result_buffer_pending’, ‘result_layer_6_thickness’, ‘teracota_heatsink_tempC’, ‘result_layer_4_thickness’}

Optional Fields

The following variables are optional in the MISO field : {‘result_alignment_status’, ‘result_transform_status’, ‘result_transformed_axis_6’, ‘result_transformed_axis_1’, ‘result_has_axis_3’, ‘green_laser_status’, ‘result_transformed_axis_3’, ‘result_has_axis_1’, ‘job_running’, ‘result_has_axis_2’, ‘result_transformed_axis_4’, ‘result_transformed_axis_5’, ‘result_transformed_axis_2’, ‘green_laser_access’}

Read offset

Empirically, we have found that some combination of hardware and firmware bugs can cause an offset in the above byte index when data is transmitted. An offset has not been observed for the current setup but we recommend that the PLC interfacing code could handle the case where this offset is non-zero.

TeraCota Status (teracota_status)

Type=uint8

This field corresponds to the current state that TeraCota is in (see Figure 1 for state diagram)

Bit #	Description
0 (LSB)	0=OFF, 1=INITIALISING, 2=READY, 3=SCANNING, 4=MEASURING, 5=STANDBY, 6=CLOSING, 7=ERROR
1	
2	
3	
4	Reserved
5	Reserved
6	Reserved
7 (MSB)	Reserved

TeraCota Heartbeat (teracota_heartbeat)

Type=uint8

This field is a heartbeat updates every 2s to signify that there is a connection to the instrument. The value will alternate between 0 and 1.

TeraCota Error Code (teracota_error_code)

Errors are divided into two categories: faults and ephemerals. Faults are always associated with “TeraCota Status = Error” and provide additional information about the fault. These faults can be cleared by reinitialising the instrument. The ephemeral category is related to the *last measurement only* and gives additional information on the quality of the measurement independent of the Result. For example, if “TeraCota Error Code = Alignment Low Signal” it is possible for the control program to move the robot to an improved location and repeat the measurement point.

The variable is updated when “TeraCota Status” makes the automatic transition from the “Measuring” state (to either “Scanning” or “Error”, see State Machine Diagram). Upon receiving any subsequent commands, “TeraCota Error Code” is reset to value 0. Therefore check “TeraCota Error Code” when measurement completes and before sending any other commands. Or if “TeraCota Status = Error” check “TeraCota Error Code” before sending the “Reinitialise” command to get more information about the fault condition.

Please see the table below for how to recover from errors.

Value	Description	Category	Recovery
0	Undefined		
1	Alignment Fault Or Alignment Unstable Signal	Ephemeral	Improve position, try again, ignore
2	Alignment Out of Range	Ephemeral	Improve position, try again, ignore
3	Alignment Low Signal	Ephemeral	Improve position, try again, ignore
4	Alignment Incomplete	Ephemeral	Improve position, try again, ignore
5	Measurement time exceeded	Ephemeral	Improve position, try again, ignore
6	Measurement iterations exceeded	Ephemeral	Improve position, try again, ignore
7	Measurement waveforms exceeded	Ephemeral	Improve position, try again, ignore
8	Paint code not set	Ephemeral	Set paint code and try again
9	Internal waveforms not arriving	Fault	Reinitialise
10	Internal state error	Fault	Reinitialise
11	Internal timeout error	Fault	Reinitialise
12	Internal check failed	Fault	Reinitialise
13	Temperature Error	Fault	Reinitialise or Intervention Needed
14	Reserved	TBD	TBD
15	Catch-all	Fault or Ephemeral	Reinitialise or Intervention Needed if also “TeraCota Status = ERROR”, otherwise, a “Failed Measurement”. Attempt to improve the robot’s position or use a different Calibration.

TeraCota Heatsink TempC (teracota_heatsink_tempC)

Type=uint8

The temperature, in degrees Celsius, of the heatsink. The system will automatically enter standby state if this goes above 45C. A value of zero is undefined, not 0 degrees Celsius.

Result Buffer Size (result_buffer_size)

Type=uint8

The number of results stored in the buffer waiting to be retrieved. A result is a set of multilayer thicknesses from a single location. The Buffer can store 255 results. If buffer overflows the oldest result will be deleted and the data will be lost.

Result Buffer Pending (result_buffer_pending)

Type=uint8

The number of measurements being processed. Upon completion the result will be added to the result buffer. Size of zero indicates the system has finished processing all measurement data.

Result Job ID (result_job_id)

Type=uint16

The Job ID corresponding to the current result.

Result Vehicle ID (result_vehicle_id)

Type=uint32

The Vehicle ID corresponding to the current result. Note that this will only contain integer values. In other words it will not contain any other meta data e.g. characters.

Result Location ID (result_location_id)

Type=uint16

The Location ID for the current result.

Result Body ID (result_body_id)

Type=uint16

The Body ID for the current result.

Result Status (result_status)

Type=uint16

This field corresponds to status of the latest requested result.

Accessing results is independent of TeraCota control state machine and consult the Result Access section for more information.

A value of 7 indicates that TeraCota is in the process of updating the MISO with new values. The control program should monitor Result Status = 7 (UPDATING) and only read results when Result Status < 7 (see Figure 2). Control program should timeout after 2s. Successful results should have Result Status = 0 (“OK”). Values 1 to 5 indicate processing failures of different types. Value 6 indicates no result exists in the buffer.

Bit#	Description
0 (LSB)	0=OK, 1=FAILED, 2=INVALID, 3=ABORTED, 4=SOME_PROBLEMS, 5=FAILED_MEASUREMENT, 6=NO_RESULT, 7= UPDATING
1	
2	
3	
4	Reserved
5	Reserved
6	Reserved
7 (MSB)	Reserved

Result Layer N Thickness (result_layer_n_thickness)

Type=uint16

The thickness of the layer N (where N ranges from 1 to 6) encoded in 100s of nanometres. Layer 1 is the layer furthest away from the substate. In total the fieldbus interface can return a maximum of 6 layers.

All thicknesses and thickness uncertainties are in units of 100s of nanometres. For example, 1 = 100 nanometres = 0.1 micrometres. Maximum thickness representable (0.1 micrometres x 65536 = 6553.6 micrometres) should be treated as Not-a-Number.

Result Layer N Uncertainty (result_layer_n_uncertainity)

Type=uint16

The uncertainty of layer N encoded in 100s of nanometres. For example, value of 10 would mean the uncertainty is ±1 micrometre.

Result Layer N Status (result_layer_n_status)

Type=uint8

A status code for the layer thickness calculation. Layer Status = 5 indicates that the layer values should be ignored entirely. For example, in the case of a 4-layer build, the Layer Status will be IGNORE (5) for the 5^{th} and 6^{th} layers.

Bit#	Description
0 (LSB)	0=VALID, 1=NOT_FINITE, 2=OUT_OF_RANGE, 3=UNCERTAIN, 4=UNKNOWN, 5=IGNORE
1	

Bit#	Description
2	
3	
4	Reserved
5	Reserved
6	Reserved
7 (MSB)	Reserved

Result Alignment Status (result_alignment_status)

Type=uint8

The HAS alignment status for the last measurement.

Bit#	Description
0 (LSB)	0=ALIGNED, 1=FAULT, 2=OUT OF RANGE, 3=LOW SIGNAL, 4=INCOMPLETE, 5=UNKNOWN
1	
2	
3	
4	Reserved
5	Reserved
6	Reserved
7 (MSB)	Reserved

Result HAS Axis (result_has_axis_n)

Type=sint16

The values 1-3 correspond to the position of the Head Alignment System (HAS). Adopting the TeraCota nomenclature this would be the yaw, pitch and focus axes for values 1,2 and 3 (although this order is configurable). To the values reported in these variables are the original HAS values multiplied by 1000 (so millidegrees for rotation axes and micrometres for translational axes). Note that this is signed as it can be positive or negative.

Result Transform Status (result_transformed_status)

Type = sint8

A status bit to describe the success of transformation to HAS to global co-ordinates.

Bit#	Description
0 (LSB)	0=SUCCESS, 1=FAILED
1	
2	
3	
4	Reserved

Bit#	Description
5	Reserved
6	Reserved
7 (MSB)	Reserved

Result Transformed Axis (result_transformed_axis_n)

Type = sint16

The result from the transformation form HAS co-ordinates to a global co-ordinate system. See Result HAS Axis for more information on units. Note that this is signed as it can be positive or negative. There will be 6 items for this field.

Green Laser Status (green_laser_status)

Type=uint8

The current state of the green alignment laser.

Bit #	Description
0	0 = OFF , 1 = ON, 2 = ERROR

Green Laser Access (green_laser_access)

Type=uint8

Whether or not the green laser can be accessed. To be access the GUI must be in Teach mode and the TeraCota scanner must be running

Bit #	Description
0	0 = NO_ACCESS, 1 = ACCESS

Job Running (job_running)

Type=uint8

Whether or not a job is running.

Bit #	Description
0	0 = NO_JOB, 1 = JOB_RUNNING

Fieldbus Output

TeraCota saves a CSV file which summarises the fieldbus configuration settings it was using. This CSV file is saved in the same location as the fieldbus configuration JSON file that is defined in the YAML. The filename will take the be the same as the configration JSON fill but the .json suffix will be replaced with .csv. An example output file is shown below.

```
formatters
measurement_name_format,['vehicle_id', 'job_id', 'location_id', 'body_id']
filename_format,['job_id']
vehicle_id_format,['_vehicle_id']
flags
_use_fieldbus_job_id,True
measurement_meta_data_map
job_id,job_id
vehicle_id,vehicle_id
body_id,body_id
location_id,location_id
MOSI
Byte Index,Python Variable Name, Size,Comment
0,_teracota_control,1,TeraCota Control
1,_measurement_type,1,Measurement Type
2,_fieldbus_job_id,2,Job ID
4,_vehicle_id,4,Vehicle ID
8,location_id,2,Location ID
10,body_id,2,Body ID
12,_paint_code_1,1,Paint Code Char1
13,_paint_code_2,1,Paint Code Char2
14,_paint_code_3,1,Paint Code Char3
15,_result_access,1,Result Access
16,spacer1,61,Spacing before second instrument
77,_teracota_control,1,TeraCota Control
78,_measurement_type,1,Measurement Type
79,_fieldbus_job_id,2,Job ID
81,_vehicle_id,4,Vehicle ID
85,location_id,2,Location ID
87,body_id,2,Body ID
89,_paint_code_1,1,Paint Code Char1
90,_paint_code_2,1,Paint Code Char2
91,_paint_code_3,1,Paint Code Char3
92,_result_access,1,Result Access
93,spacer1,61,Spacing before second instrument
MISO
Byte Index,Python Variable Name, Size,Comment
0,teracota_status,1,TeraCota Stasus
1,teracota_heartbeat,1,TeraCota Heartbeat
2,teracota_error_code,1,TeraCota Error Code
3,teracota_heatsink_tempC,1,TeraCota Heatsink TempC
4,result_buffer_size,1,Results Buffer Size
5,result_buffer_pending,1,Results Buffer Pending
6,result_job_id,2,Results Job ID
8,result_vehicle_id,4,Results Vehicle ID
12,result_location_id,2,Results Location ID
14,result_body_id,2,Results Body ID
16,result_status,2,Results Status
18,result_layer_1_thickness,2,Results Layer 1 Thickness
20,result_layer_2_thickness,2,Results Layer 2 Thickness
22,result_layer_3_thickness,2,Results Layer 3 Thickness
24,result_layer_4_thickness,2,Results Layer 4 Thickness
26,result_layer_5_thickness,2,Results Layer 5 Thickness
28,result_layer_6_thickness,2,Results Layer 6 Thickness
30,result_layer_1_uncertainty,2,Results Layer 1 Uncertainty
32,result_layer_2_uncertainty,2,Results Layer 2 Uncertainty
34,result_layer_3_uncertainty,2,Results Layer 3 Uncertainty
36,result_layer_4_uncertainty,2,Results Layer 4 Uncertainty
38,result_layer_5_uncertainty,2,Results Layer 5 Uncertainty
40,result_layer_6_uncertainty,2,Results Layer 6 Uncertainty
42,result_layer_1_status,1,Results Layer 1 Status
43,result_layer_2_status,1,Results Layer 2 Status
44,result_layer_3_status,1,Results Layer 3 Status
45,result_layer_4_status,1,Results Layer 4 Status
46,result_layer_5_status,1,Results Layer 5 Status
47,result_layer_6_status,1,Results Layer 6 Status
```

```
48,result_alignment_status,2,Results HAS Alignment
50,result_has_axis_1,2,Results HAS Axis 1
52,result_has_axis_2,2,Results HAS Axis 2
54,result_has_axis_3,2,Results HAS Axis 3
56,result_transform_status,1,Results Transform Status
57,result_transformed_axis_1,2,Results Transformed Axis 1
59,result_transformed_axis_2,2,Results Transformed Axis 2
61,result_transformed_axis_3,2,Results Transformed Axis 4
63,result_transformed_axis_4,2,Results Transformed Axis 4
65,result_transformed_axis_5,2,Results Transformed Axis 5
67,result_transformed_axis_6,2,Results Transformed Axis 6
69,spacer1,22,Spacing before second instrument
91,teracota_status,1,TeraCota Stasus
92,teracota_heartbeat,1,TeraCota Heartbeat
93,teracota_error_code,1,TeraCota Error Code
94,teracota_heatsink_tempC,1,TeraCota Heatsink TempC
95,result_buffer_size,1,Results Buffer Size
96,result_buffer_pending,1,Results Buffer Pending
97,result_job_id,2,Results Job ID
99,result_vehicle_id,4,Results Vehicle ID
103,result_location_id,2,Results Location ID
105,result_body_id,2,Results Body ID
107,result_status,2,Results Status
109,result_layer_1_thickness,2,Results Layer 1 Thickness
111,result_layer_2_thickness,2,Results Layer 2 Thickness
113,result_layer_3_thickness,2,Results Layer 3 Thickness
115,result_layer_4_thickness,2,Results Layer 4 Thickness
117,result_layer_5_thickness,2,Results Layer 5 Thickness
119,result_layer_6_thickness,2,Results Layer 6 Thickness
121,result_layer_1_uncertainty,2,Results Layer 1 Uncertainty
123,result_layer_2_uncertainty,2,Results Layer 2 Uncertainty
125,result_layer_3_uncertainty,2,Results Layer 3 Uncertainty
127,result_layer_4_uncertainty,2,Results Layer 4 Uncertainty
129,result_layer_5_uncertainty,2,Results Layer 5 Uncertainty
131,result_layer_6_uncertainty,2,Results Layer 6 Uncertainty
133,result_layer_1_status,1,Results Layer 1 Status
134,result_layer_2_status,1,Results Layer 2 Status
135,result_layer_3_status,1,Results Layer 3 Status
136,result_layer_4_status,1,Results Layer 4 Status
137,result_layer_5_status,1,Results Layer 5 Status
138,result_layer_6_status,1,Results Layer 6 Status
139,result_alignment_status,2,Results HAS Alignment
141,result_has_axis_1,2,Results HAS Axis 1
143,result_has_axis_2,2,Results HAS Axis 2
145,result_has_axis_3,2,Results HAS Axis 3
147,result_transform_status,1,Results Transform Status
148,result_transformed_axis_1,2,Results Transformed Axis 1
150,result_transformed_axis_2,2,Results Transformed Axis 2
152,result_transformed_axis_3,2,Results Transformed Axis 4
154,result_transformed_axis_4,2,Results Transformed Axis 4
156,result_transformed_axis_5,2,Results Transformed Axis 5
158,result_transformed_axis_6,2,Results Transformed Axis 6
160,spacer1,22,Spacing before second instrument
```

Summary of Fieldbus Interface Control Flow

Figure 3 outlines the process for controlling a TeraCota instrument across the fieldbus. Before any PLC communications can occur the TeraCota application must first be started on the rack PC.

TeraCota collects and groups multiple measurements together into a ‘job’. Each job is defined by a Job ID which should be a unique integer that is either provided by the PLC or internally counted in TeraCota. The fact that this integer is unique to each job is crucial as TeraCota gen-

erates output files based on this Job ID. If, for example, repeated Job IDs are sent by the PLC then the corresponding output files generated by TeraCota will contain measurements from all previous jobs with that Job ID. If the PLC is providing this Job ID it is therefore the responsibility of the PLC to ensure that all Job IDs are unique.

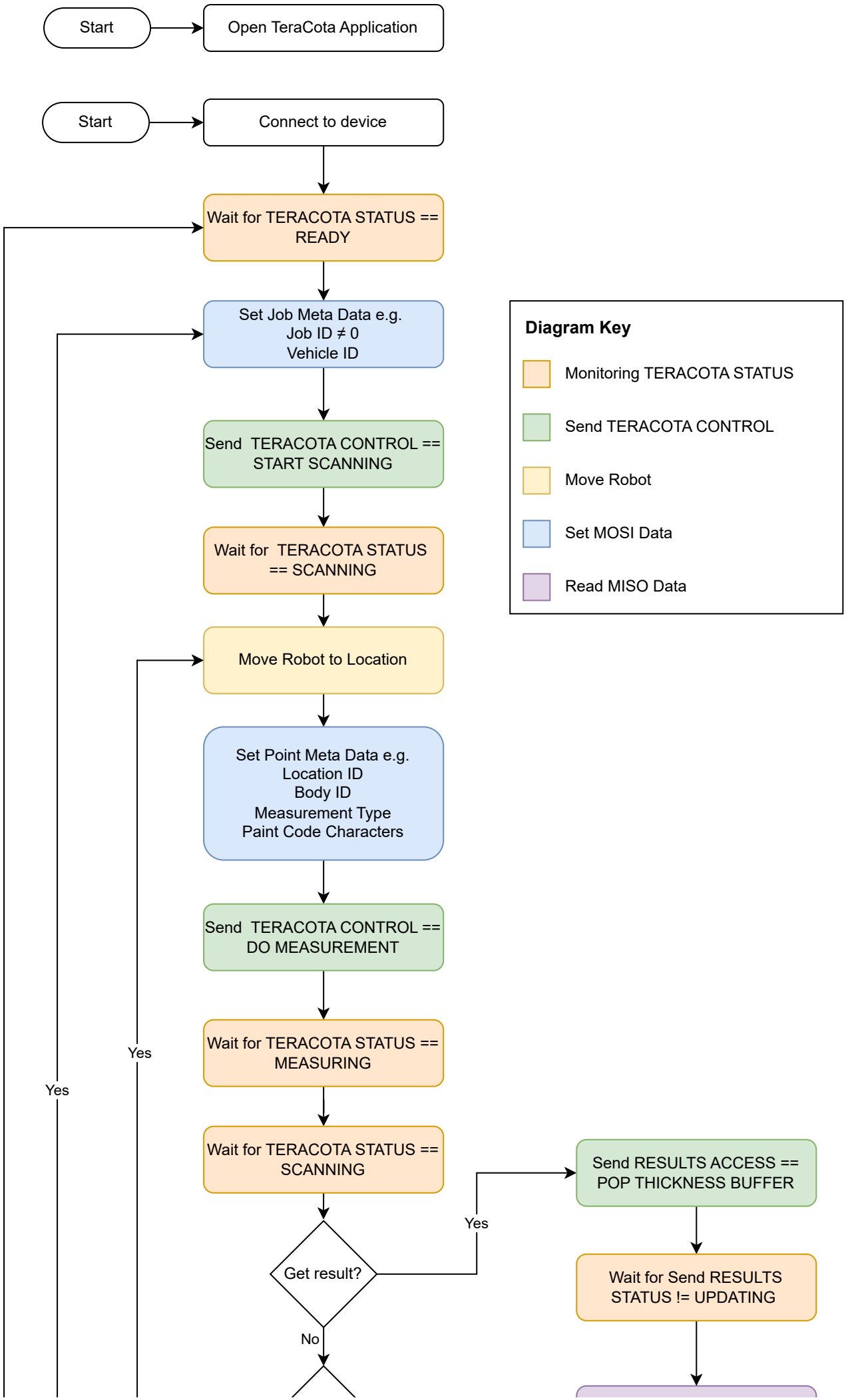
TeraCota considers a job to start when the Job ID in the MOSI data block changes from zero to a number that is non-zero (Figure 3). Note that the vehicle ID and other meta data this is contant for a job must also be set here before the scanner is started. Other meta data values (such as Location ID and Body ID) are set before each measurement. TeraCota considers a job to be finished when the Job ID in the MOSI data block changes from a non-zero number to zero. Once this occurs TeraCota proceeds to create output files for the Job ID that has just finished. If a TeraCota controlled Job ID is to be used then the same process must still be followed, but in this case TeraCota will not assign the numerical value of Job Id to the job in question (instead it will use the internal Job ID value).

In the case of a dual instrument system the instrument that corresponds to sensor #1 (see MOSI data structure table) is the lead instrument in terms of setting Job ID. Therefore, any changes to the Job ID should only be applied to sensor #1. The numerical value set here will be shared between instruments. Moreover, only changes in the Job ID for sensor #1 will be monitored in order to trigger the start and end of jobs.

Allow 200ms before setting any data and expecting a response from TeraCota. TeraCota refreshes Fieldbus Master data every 100ms.

If the system is not being used for a several hours (the exact time is configurable), it is recommended to follow the sequence to enter STANDBY state (See TeraCota state machine).

TeraCota will automatically enter the STANDBY state if DO MEASUREMENT has not occurred in a user configured timeout period. Typically this period is set to 2 hours and is configured in the TeraCota configuration file.


Start
Open TeraCota Application
Start
Connect to device
Wait for TERACOTA STATUS == READY
Set Job Meta Data e.g.
Job ID ≠ 0
Vehicle ID
Send TERACOTA CONTROL == START SCANNING
Wait for TERACOTA STATUS == SCANNING
Move Robot to Location
Set Point Meta Data e.g.
Location ID
Body ID
Measurement Type
Paint Code Characters
Send TERACOTA CONTROL == DO MEASUREMENT
Wait for TERACOTA STATUS == MEASURING
Wait for TERACOTA STATUS == SCANNING
Get result?
Yes
No
Send RESULTS ACCESS == POP THICKNESS BUFFER
Wait for Send RESULTS STATUS != UPDATING
Yes
Yes
Diagram Key
Monitoring TERACOTA STATUS
Send TERACOTA CONTROL
Move Robot
Set MOSI Data
Read MISO Data

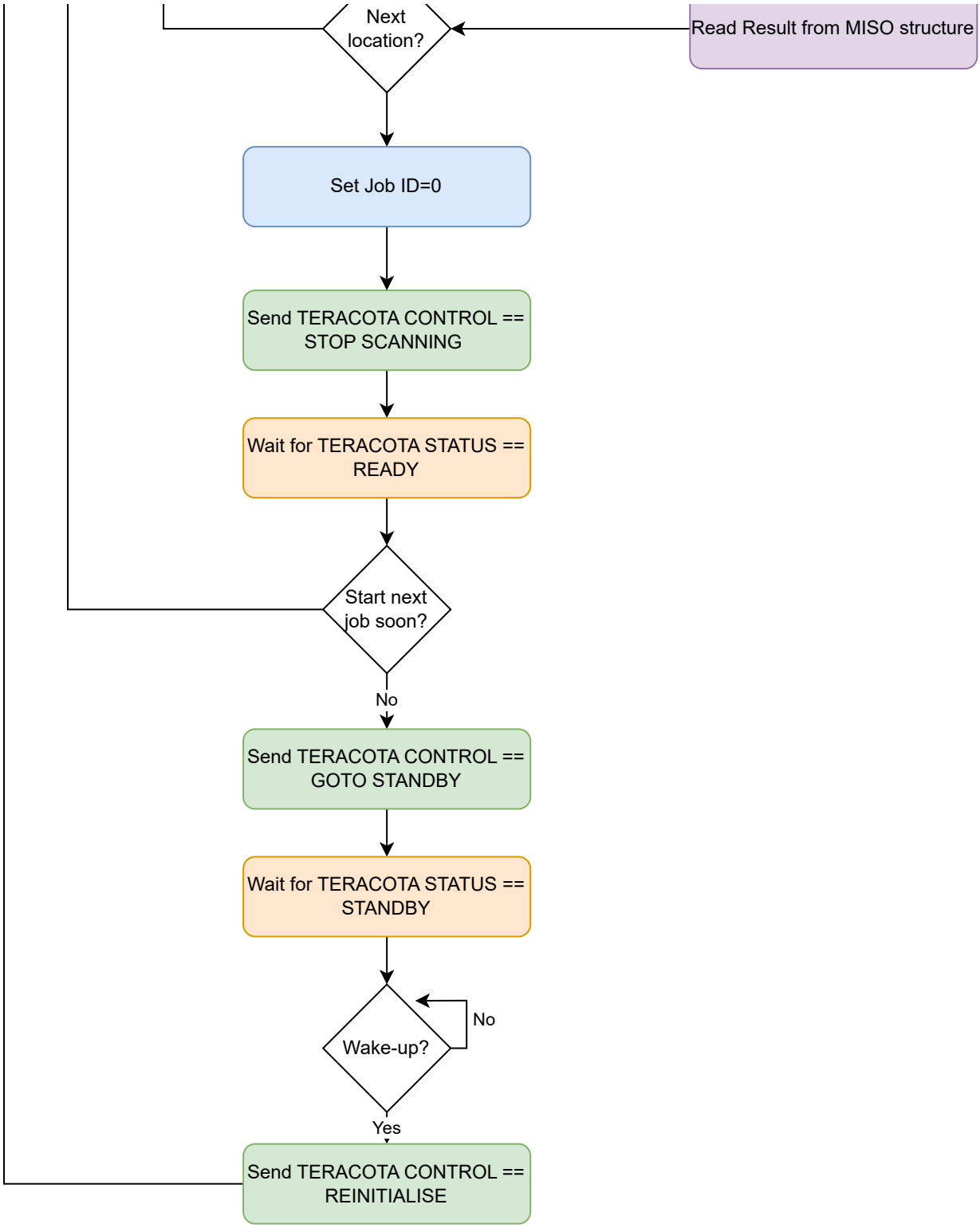
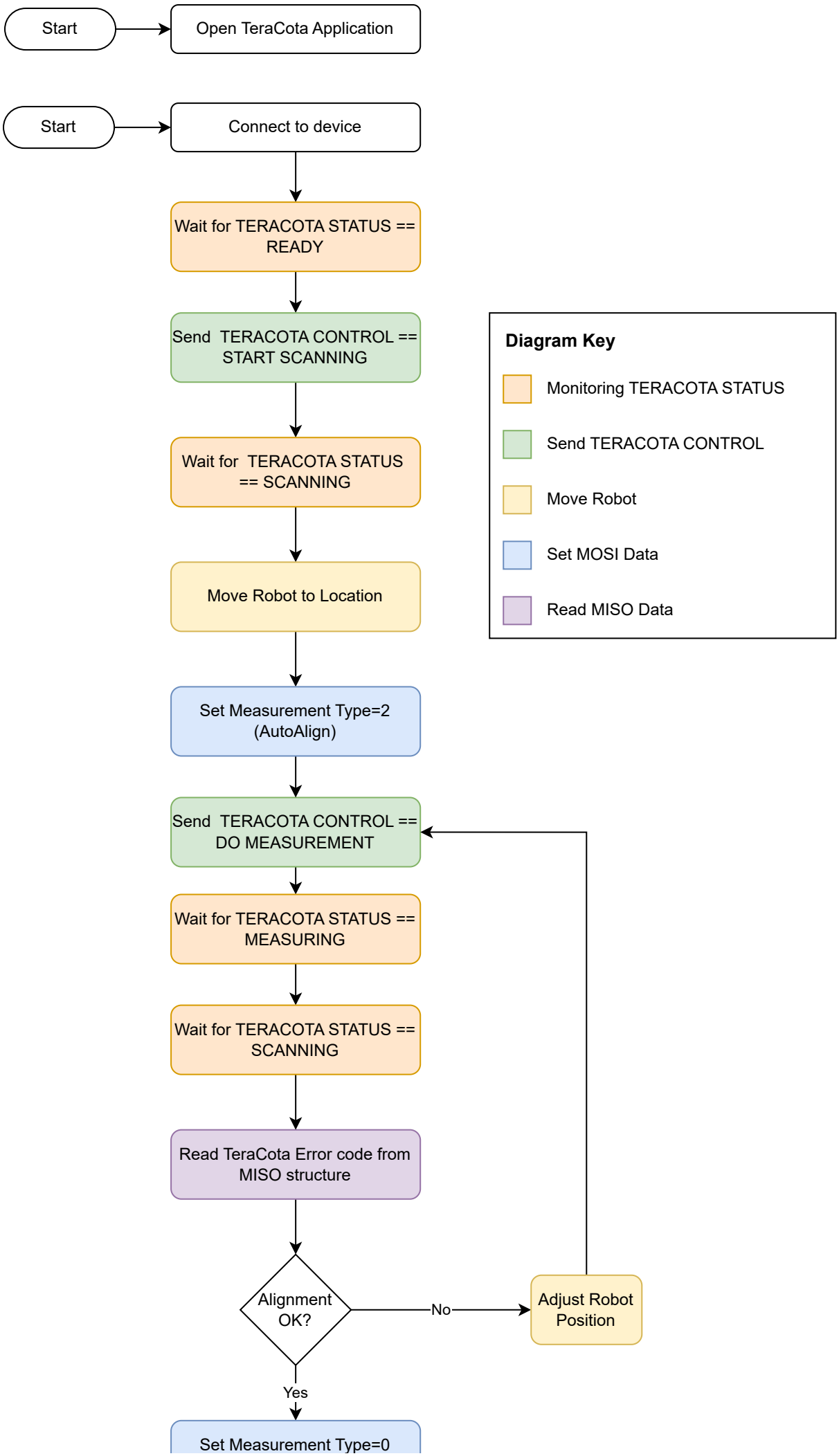


Figure 3 : Fieldbus Flow Control

Example case with AutoAlign and Point Measurements

In most cases using the AutoAlignPoint measurement type is the best approach as this will ensure the alignment of the sensor to the surface of the sample is optimised. However, this will result in the longest acquisition time . Another approach is to perform one AutoAlign measurement followed by multiple Point measurements. Thus the acquisition time is reduced relative to the AutoAlignPoint approach as the alignment is only completed once. This is a compromise as the alignment cannot be guaranteed to be optimal for all Point measurements. An example control flow for the AutoAlign and Point approach is given in Figure 4 below. For brevity metadata, MISO reads, standby and Job ID control are not included in Figure 4.


Start
Open TeraCota Application
Start
Connect to device
Wait for TERACOTA STATUS == READY
Send TERACOTA CONTROL == START SCANNING
Wait for TERACOTA STATUS == SCANNING
Move Robot to Location
Set Measurement Type=2 (AutoAlign)
Send TERACOTA CONTROL == DO MEASUREMENT
Wait for TERACOTA STATUS == MEASURING
Wait for TERACOTA STATUS == SCANNING
Read TeraCota Error code from MISO structure
Alignment OK?
No
Adjust Robot Position
Yes
Set Measurement Type=0
Diagram Key
Monitoring TERACOTA STATUS
Send TERACOTA CONTROL
Move Robot
Set MOSI Data
Read MISO Data

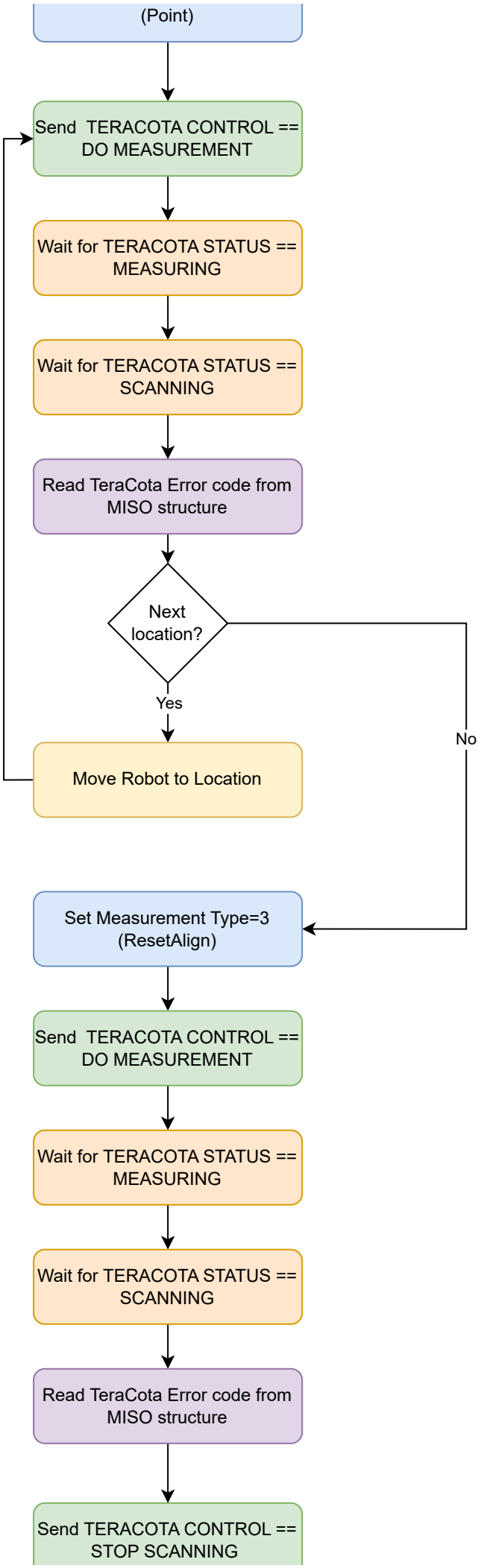

(Point)
Send TERACOTA CONTROL == DO MEASUREMENT
Wait for TERACOTA STATUS == MEASURING
Wait for TERACOTA STATUS == SCANNING
Read TeraCota Error code from MISO structure
Next location?
Yes
No
Move Robot to Location
Set Measurement Type=3 (ResetAlign)
Send TERACOTA CONTROL == DO MEASUREMENT
Wait for TERACOTA STATUS == MEASURING
Wait for TERACOTA STATUS == SCANNING
Read TeraCota Error code from MISO structure
Send TERACOTA CONTROL == STOP SCANNING

Figure 4 : Example Flow Control with AutoAlign and Point Measurements