ACS

Testing Tools

From version 2.9

Part Number: ACS_TT_2.9_EN

Ubisense Limited, St Andrew's House, St Andrew's Road, Cambridge CB4 1DL, United Kingdom.

Telephone: +44 (0)1223 535170. Website: http://www.ubisense.net.

Ubisense Limited
St Andrew's House
St Andrew's Road
Cambridge CB4 1DL
United Kingdom

Tel: +44 (0)1223 535170

WWW: *https://www.ubisense.com*

Contents

Introduction

ACS provides a set of command-line tools to help people responsible for the configuration of ACS with testing their installation:

- *ubisense_product_tag_simulator*
- *ubisense_device_tag_simulator*
- *ubisense_acs_protocol_clientserver*
- *ubisense_open_protocol_protocol_clientserver*

To download the tools from a running ACS dataset:

1. Open the **DOWNLOADABLES** tab in Ubisense Application Manager.
2. Navigate to **Ubisense Generation 2.X** > **ACS** and select **Tools**.
3. Click **Download selected items**.

ubisense_product_tag_simulator

The **ubisense_product_tag_simulator** tool can be used to simulate tags moving along tag-driven assembly lines. If the tags are associated with products, tracking support services will pick up the tag locations and move the associated products along the assembly lines. Moving products can then be used to generate various events, such as: products entering or leaving stations, workspaces or ident zones; or products passing trigger points. The events can finally lead to telegrams being sent to external systems associated with the events.

Run **ubisense_product_tag_simulator(.exe)** at the command prompt in a shell or a Command Prompt window without any parameters to get the list of available options:

```
ubisense_product_tag_simulator.exe
12.03.2021 08:43:32.954 [M] Set LogLevel to 'Information'
Not enough parameters given. Missing: ProductionLine.

Usage: ubisense_product_tag_simulator.exe [OPTIONS] <ProductionLine>
Generates tag location updates on ACS assembly lines

PARAMETERS
  ProductionLine            The name of the production line for which tag
                            locations shall be generated


OPTIONS
  -l, --log-level <log_level>
                            The log level regarding log output (Debug,
                            Information, Warning, Error)

  -f, --start-filled        If the flag is set the simulation will fill the
                            lines with tags upfront

  -t, --start-tag-id <start_tag_id>
                            The first tag ID used by the simulation (default
                            000-000-000-001)

  -u, --upper-bound-tag-id <upper_bound_tag_id>
                            The highest tag ID to be used

  -z, --z-offset <z_offset> The offset in z direction added to the tag
                            position (default = 1.0)

  --help                    Display this help and exit

  --version                 Display version information and exit
```

For example:

```
ubisense_product_tag_simulator.exe -t 0011CE0000000F00 -u 0011CE0000000F1F MainLine
```

This command starts the simulation for a production line called 'MainLine' with tags starting at tag ID 0011CE0000000F00. When the simulation starts, the first tag is positioned at the beginning of the first assembly line and then moves along the line at the speed configured for the assembly line. As soon as the tag has moved 'minimum carrier distance' configured for the assembly line, the next tag is positioned at the beginning of the assembly line. Its tag ID is the former tag ID incremented by 1, in this example 0011CE0000000F01. This process continues until all assembly lines of the production line are full of tags. At that point the location of the tag with the ID 0011CE0000000F00 will be removed. Once all tags up to the ID 0011CE0000000F1F have been used, the simulator will place tag ID 0011CE0000000F00 at the beginning of the first assembly line again.

When tags reach the end of an assembly line, their locations are removed. If there is a subsequent assembly line in the production line, they are then located at the beginning of the next assembly line and move there at the speed configured for this next assembly line. If the speed is slower than the speed of the previous assembly line, up to 3 tags will be put into a buffer between the assembly lines. Once this buffer is full, the previous assembly line will no longer be able to deliver tags into it, and stops until the buffer can accept another tag. This will be possible as soon as the subsequent assembly line has picked the next tag out of the buffer. If the speed of the subsequent assembly line is higher than the speed of the previous assembly line, there may be no tags available because the buffer between the lines is empty. In this case the second assembly line will carry on moving the tags it already has, with the result that tag distances on the line are bigger than the configured 'minimum carrier distance'.

The ProductionLine Parameter

The ProductionLine parameter is mandatory. It must be the name of a Production Line configured in ACS that has at least one tag-driven assembly line associated with it. Production lines with associated externally-driven assembly lines are not supported by the **ubisense_product_tag_simulator** tool.

Tag locations will be simulated along the associated assembly lines, in the sequence configured for the production line.

Options

log-level

The log level option requires one of the following values to be set: Debug, Information, Warning, Error. Depending on the log level, the amount of data the simulator writes into a log file will change. The default log level is 'Information', which should be sufficient for most cases. Use 'Debug' to investigate and understand why the simulator behaves as it does.

start-filled

This option is a flag and does not require further input. If the flag is set, the simulation will fill the lines with tags upfront, i.e. there will be tag locations all along all assembly lines. This option makes sense if the tags are already associated with products.

start-tag-id

This option requires a <start_tag_id> to be specified. The tag ID can be given in hexadecimal form (e.g. 0011CE0000000100), in hexadecimal form with colon separators (e.g. 00:11:CE:00:00:00:01:00) or in the octet form (e.g. 100-001-002-000). The tag with the <start_tag_id> will be the first tag being located. This tag ID will then be incremented by 1 for all subsequent tags.

The default start-tag-id is '1'.

upper-bound-tag-id

This option requires a <upper_bound_tag_id> to be specified. The format is the same as described for the option start-tag-id. The simulator will increment tag IDs until the <upper_bound_tag_id> is reached. Then it will fall back to the <start_tag_id>. However, if the number of tags necessary to fill all assembly lines with tags is greater than <upper_bound_tag_id> - <start_tag_id>, the tag IDs will be further incremented until the tag with ID <start_tag_id> leaves the last assembly line.

Hint: You can specify the upper-bound-tag-id just as start-tag-id + 1, in which case the simulator will just use as many tag IDs as required to fill all assembly lines.

z-offset

This option requires a <z_offset> to be specified in meters. The height of the tag position will be the height of the assembly line plus this value. The default value for <z_offset> is 1.0 m.

ubisense_device_tag_simulator

The **ubisense_device_tag_simulator** tool can be used to simulate tags moving inside assembly line stations. For each station of a production line's assembly lines, tags will be moving inside the extent of the station. If the tags are associated with devices, the devices may move into or out of product spaces for products moving along the assembly lines and by that means generate device events. The events can finally lead to telegrams being sent to external systems associated with the events.

Run **ubisense_device_tag_simulator(.exe)** at the command prompt in a shell or a Command Prompt window without any parameters to get the list of available options:

```
ubisense_device_tag_simulator.exe
12.03.2021 13:36:40.910 [M] Set LogLevel to 'Information'
Not enough parameters given. Missing: ProductionLine.

Usage: ubisense_device_tag_simulator.exe [OPTIONS] <ProductionLine>
Generates tag location updates in ACS assembly line stations

PARAMETERS
  ProductionLine            The name of the production line for which tag
                            locations shall be generated


OPTIONS
  -i, --interval <interval> The number of seconds between each update of the
                            simulation (default 1s)

  -l, --log-level <log_level>
                            The log level regarding log output (Debug,
                            Information, Warning, Error)

  -t, --start-tag-id <start_tag_id>
                            The first tag ID used by the simulation (default
                            00:00:00:01)

  -s, --tag-speed <tag_speed>
                            The global speed of device tags (default 0.25 m/s)

  -z, --z-offset <z_offset> The offset in z direction added to the tag
                            position (default = 1.5)

  --help                    Display this help and exit

  --version                 Display version information and exit
```

For example:

```
ubisense_device_tag_simulator.exe -t 0011CE0000AB0F00 MainLine
```

This command would start the simulation for the production line called 'MainLine' and with tags starting with the tag ID 0011CE0000AB0F00. The number of simulated tags will be equal to the total number of stations associated with assembly lines of the production line.

When the simulation starts, the tag with ID 0011CE0000AB0F00 will be positioned inside the first station of the first assembly line. The tag with an ID incremented by 1 will be positioned inside the second station of the first assembly line. This will be repeated until tags are positioned in all stations of all assembly lines.

The simulation will then move the tags inside their stations, within the extent of the station. If devices are associated with the tags, the tag movements will move the devices.

ProductionLine Parameter

The ProductionLine parameter is mandatory. It must be the name of a Production Line as configured in ACS.

Tag locations will be simulated within the stations of the assembly lines associated with the production line.

Options

interval

This option requires the value <interval> to be set. The value configures the number of seconds between each update of the simulation (default 1s). The value can be fractional, so that intervals of less than one second can be specified.

The simulator will update tag locations with an update rate as configured for the tag. For DIMENSION4 software installations, where tag update rates are not configurable through the platform, the tag update rate uses the default of 32, which is approximately 1 Hz.

In a future implementation there may be an additional option to set the update rate for DIMENSION4 software tags.

log-level

The option log level requires one of the following values being set: Debug, Information, Warning, Error. Depending on the log level the amount of data the simulator writes into a log file will change. The default log level is 'Information', which should be OK in most of the cases. Use 'Debug' to investigate and understand why the simulator behaves as it does.

start-tag-id

This option requires a <start_tag_id> to be specified. The tag ID can be given in hexadecimal form (e.g. 0011CE0000000100), in hexadecimal form with colon separators (e.g. 00:11:CE:00:00:00:01:00) or in the octet form (e.g. 100-001-002-000). The tag with the <start_tag_id> will be the first tag being located. This tag ID will then be incremented by 1 for all subsequent tags.

The default start-tag-id is '1'.

tag-speed

This option requires a <tag_speed> value to be specified. The value is the speed with which tags move inside the stations. The value can be fractional, the unit is m/s. Default value is 0.25m/s.

z-offset

This option requires an <z_offset> to be specified, the unit is meters. The height of the tag position will be the height of the assembly line plus this value. The default value for <z_offset> is 1.5m.

ubisense_acs_protocol_clientserver

The **ubisense_acs_protocol_clientserver** simulates an external system implementing the ACS Protocol. The simulator can be connected to an external system configured in ACS and be used to send or receive data.

The tool is used in development to simulate specific situations, but can be used in test as well, specifically to simulate

- an event receiver, such as a tool or stationary system receiving device or product events
- an association station delivering production numbers to ACS
- an MES delivering additional data for products to ACS
- a PLC delivering production numbers and product positions for externally driven assembly lines

Start the ubisense_acs_protocol_clientserver(.exe) on the command line in a shell or a command window without any parameters to get the list of available options:

```
ubisense_acs_protocol_clientserver.exe
12.03.2021 14:12:41.471 [M] Set SynchronousWrite to '1'
12.03.2021 14:12:41.487 [M] Set Cout to '1'
Usage: ubisense_acs_protocol_clientserver.exe server <port> <testcase>
   or: ubisense_acs_protocol_clientserver.exe server <server> <port> <testcase>
   or: ubisense_acs_protocol_clientserver.exe server <server> <port> <testcase>
<client to accept>
   or: ubisense_acs_protocol_clientserver.exe client <server> <port> <testcase>
   or: ubisense_acs_protocol_clientserver.exe [client|server] <server> <port> 1
<ResponseDelay[s]>
   or: ubisense_acs_protocol_clientserver.exe [client|server] <server> <port> 14
AssociationZone ProductTypes[ColonSeparated] FirstProductID[numeric] ProductIDPrefix
[(empty)string]
   or: ubisense_acs_protocol_clientserver.exe [client|server] <server> <port> 16
LineName #Products StartOffset[cm] ProductDistance[cm] ProductTypes[ColonSeparated]
FirstProductID[numeric] LineSpeed[m/s] CycleDelayTime[s]
   or: ubisense_acs_protocol_clientserver.exe [client|server] <server> <port> 100
<InputFile> <Delay[s]>
Where testcase
   1: Connect, then listen
   2: Connect, then listen, but do not answer
   3: Connect, Disconnect
   4: Connect, send some messages
   5: Connect, server sends unknown identifier as responses to the events
   7: Connect, send messages as fast as possible
   8: Connect, then listen silently
   9: Connect, then send ProdAt telegrams interactively
  10: Connect, listen and respond to PRODAT telegrams with PRODTAG telegrams
  11: Connect, then send PRODDTRQ telegrams interactively
  12: Connect, listen and respond to PRODDTRQ telegrams with PRODDATA telegrams
  13: Connect, then send PRODDATA telegrams interactively
  14: Connect, then continuously send ProdAt telegrams for simulating associations
  15: Connect, then listen, and as client dont send keep alive messages
  16: Connect, send PRODAT telegrams continuously for an externally driven assembly
line
  17: Connect, send TAGPOS telegrams continuously
  18: Connect, then send TAGSTRQ telegrams interactively
  19: Connect, then send TAGSTAT telegrams interactively
  20: Connect, then send APPCONF telegrams interactively
  21: Connect, then send EXCEPTRQ telegrams interactively
 100: Connect, Send messages from file
```

The test cases which are useful for application testing are

- 1: Connect, then listen
- 12: Connect, listen and respond to PRODDTRQ telegrams with PRODDATA telegrams
- 14: Connect, then continuously send ProdAt telegrams for simulating associations
- 16: Connect, send PRODAT telegrams continuously for an externally driven assembly line
- 100: Connect, Send messages from file

ubisense_acs_protocol_clientserver

Assembly Line PLC Simulation

Use test case 16 to simulate a line PLC which regularly sends production numbers and product positions to ACS. Such PLCs scan production numbers from barcodes attached to products when products arrive at the start of an assembly line. They then measure the movement of the assembly line and by that know at any time how far products have been moved down the line. They can then deliver the distance of the product from the beginning of the assembly to ACS.

This behavior can be simulated with test case 16.

ubisense_acs_protocol_clientserver.exe [client|server] <server> <port> 16 LineName #Products StartOffset[cm] ProductDistance[cm] ProductTypes[ColonSeparated] FirstProductID[numeric] LineSpeed[m/s] CycleDelayTime[s]

An example command with parameters and options is

ubisense_acs_protocol_clientserver.exe client 127.0.0.1 4007 16 Line45 23 100 600 X11:W12 4500001 0.2 1

In this setup the simulator would

- act as a TCP client
- connect to ACS on IP address 127.0.0.1 and port 4007
- run with testcase 16
- send data to assembly line called Line45
- provide data for 23 products
- set the smallest product offset to 100cm
- have a distance of 600cm between subsequent products
- use the product types ‘X11’ and ‘W12’ alternating
- use a production number 4500001 as the first production number
- move products at a speed of 0.2m/s
- send a PRODAT telegram every second

In the following all the different parameters for testcase 16 are described in detail:

Parameter [client|server]: In the case of a value of ‘client’ the tool will act as TCP client, in case of ‘server’ as TCP server. The value to choose depends on the configuration of the external system in ACS.

Parameters <server> and <port>: The IP parameters for the connection. If the tool is run as 'server', this must be a valid IP address of the box the tool is running on. The port number must be a free number, not used by any other program. Good candidates are numbers between 10000 and 20000.

Parameter LineName: The name of the assembly line as configured in ACS

Parameter #Products. The number of products and product positions, the tool shall deliver. This number must be choosen considering the length of the assembly line and the product distance (see below)

Parameter StartOffset[cm]: This is the smallest offset value the tool will deliver.

Parameter ProductDistance[cm]: The value describes the distance between subsequent product offsets

Parameter ProductTypes[ColonSeparated]: A colon separated list of product type names as configured in ACS. The production line must be configured to support the given product types. The tool will deliver products alternating the given product types.

Parameter FirstProductID[numeric]: The given value is the first production number the tool will use. Subsequent products will have production numbers incremented by 1.

Parameter LineSpeed[m/s]: The LineSpeed parameter describes the speed of the products on the line. The tool will generate product offsets such that the products move with the configured line speed.

Parameter CycleDelayTime[s]: The time in seconds between two subsequent telegrams sent by the tool. Note, that ACS does not update product positions in the frequency of the telegrams, but with the frequency configured in ACS for the line as 'Cycle time'.

Association station simulation

Use test case 14 to simulate a PLC, which sends product information to ACS whenever a new product appears at an association station. Such PLCs scan production numbers from barcodes sticked to products when products arrive at the start of an assembly line. At that location, an association zone can be configured in ACS, which detects tags and associates the tags to the products received from the PLC.

This behaviour can be simulated with test case 14.

ubisense_acs_protocol_clientserver.exe [client|server] <server> <port> 14 AssociationZone ProductTypes[ColonSeparated] FirstProductID[numeric] ProductIDPrefix[(empty)string]

An example command with parameters and options is

ubisense_acs_protocol_clientserver.exe server 127.0.0.1 4001 14 HL40-Assoc X11:W12 4500001 P

In this setup the simulator would

- act as a TCP server
- allow ACS to connect IP address 127.0.0.1 and port 4001
- run with testcase 14
- send data for an association zone called HL40-Assoc
- use the product types 'X11' and 'W12' alternating
- use number 4500001 as the numeric part of the first production number
- Use a prefix 'P' for all production numbers, so the first complete production number will be P4500001

In the following all the different parameters for test case 14 are described in detail:

Parameter [client|server]: In the case of a value of 'client' the tool will act as TCP client, in case of 'server' as TCP server. The value to choose depends on the configuration of the external system in ACS.

Parameters <server> and <port>: The IP parameters for the connection. If the tool is run as 'server', this must be a valid IP address of the box the tool is running on. The port number must be a free number, not used by any other program. Good candidates are numbers between 10000 and 20000.

Parameter AssociationZone: The name of the association zone to use for performing the associations. The association zone must be configured in ACS and be associated to the External System receiving the telegrams from the simulation tool.

Parameter ProductTypes[ColonSeparated]: A colon separated list of product type names as configured in ACS. The association zone must be configured to support the given product types. The tool will deliver products alternating the given product types.

Parameter FirstProductID[numeric]: The given value is the first production number the tool will use. This number may be prefixed with the ProductIDPrefix (see below). Subsequent products will have production numbers incremented by 1

Parameter ProductIDPrefix[(empty)string]: This is an optional parameter. The given prefix will be put in front of the numeric production numbers.

Event receiver simulation

Use testcase 1 to simulate an external system receiving event information from ACS, such as a tool controller or a PLC. The simulation tool will just connect to ACS, receive event telegrams, such as ISINZ, ISOUTZ or ZONESTAT and respond with necessary RESPONSE telegram.

This behaviour can be simulated with testcase 1.

ubisense_acs_protocol_clientserver.exe [client|server] <server> <port>

An example command is

ubisense_acs_protocol_clientserver.exe client 127.0.0.1 4011 1

In this setup the simulator would

- act as a TCP client
- connect to ACS on IP address 127.0.0.1 and port 4011
- run with testcase 1

There are no specific parameters for test case 1.

MES simulation

Use testcase 12 to simulate a MES (Manufacturing Execution System), which provides additional information for products. In ACS configure an External System with an Endpoint Role as 'ProductDataProvider'. Such an External System sends requests to the MES, using the PRODDTRQ telegrams, and expects the MES respond with a PRODDATA telegram, sending additional information to the product requested by the PRODDTRQ telegram.

This behaviour can be simulated with testcase 12.

ubisense_acs_protocol_clientserver.exe [client|server] <server> <port> 12

An example command with parameters and options is

ubisense_acs_protocol_clientserver.exe server 127.0.0.1 4015 12

In this setup the simulator would

- act as a TCP server
- allow ACS to connect on IP address 127.0.0.1 and port 4015
- run with testcase 12

There are no specific extra parameters to this test case.

The simulator responds each PRODDTRQ request with a PRODDATA telegram delivering the following product parameters:

SeqNo: A product sequence number, starting with a value of 1 and incremented by 1 for each further telegram

Color: A color name from a set of predefined color names

Destination: A country name from a set of predefined country names

Model: A model name from a predefined set of model names

PTSDate: The current date in a string format

Generic simulator

Use testcase 100 to run the simulator with an input file, which contains the telegrams to be sent. Additionally the file can contain control commands to control the delay between telegrams and allow to loop and send the same telegrams multiple times.

This behaviour can be simulated with testcase 100.

ubisense_acs_protocol_clientserver.exe [client|server] <server> <port> 100 <InputFile> <Delay [s]>

An example command with parameters and options is

ubisense_acs_protocol_clientserver.exe server 127.0.0.1 4099 100 sample.in 3

In this setup the simulator would

- act as a TCP server
- allow ACS to connect on IP address 127.0.0.1 and port 4099
- run with testcase 100
- use commands from a file called sample.in
- use an initial delay of 3 seconds between subsequent telegrams.

In the following all the different parameters for test case 100 are described in detail:

Parameter [client|server]: In the case of a value of ‘client’ the tool will act as TCP client, in case of ‘server’ as TCP server. The value to choose depends on the configuration of the external system in ACS.

Parameters <server> and <port>: The IP parameters for the connection. If the tool is run as ‘server’, this must be a valid IP address of the box the tool is running on. The port number must

be a free number, not used by any other program. Good candidates are numbers between 10000 and 20000.

Parameter <InputFile>: The name of an input file containing telegrams to be sent and control commands for specifying telegram sending delays and loops. An example input file is described below.

Parameter <Delay[s]>: This is an optional parameter, specifying the time in seconds between subsequent telegrams. This is an initial time which can be overridden by the #delay command in the input file.

Input file example:

TH0100000022***ALIVE01

#do

#delay 5

TH0100000054UNSUBSCR01**********MyDeviceSubscription

TH0100000224**SUBSCR01**********MyDeviceSubscription010001**********Vehicle**********Bonnet0001**********MyDevice**********PowerFocus

#delay 10

TH0100000022***ALIVE01

#repeat 1000

The #delay command allows specifying the time between subsequent telegrams. The number behind the #delay specifies the delay in seconds

The #do #repeat command pair allows specifying a loop. The number behind the #repeat specifies how often the loop is performed. All the telegrams between #do and #repeat are sent repeatedly.

In the example above

- An ALIVE telegram is sent
- A loop start is configured
- The delay is set to 5 seconds
- AN UNSUBSCR telegram is sent
- A SUBSCR telegram is sent

- The delay is set to 10 seconds
- An ALIVE telegram is sent
- The loop end is configured, such as the commands within the loop are executed 1000 times.

ubisense_open_protocol_clientserver

The **ubisense_open_protocol_clientserver** acts as a simulator of an external system implementing the Atlas Copco Open Protocol. The simulator can be connected to an external system configured in ACS and be used to receive data.

The tool is used in development to simulate specific situations, but can be used in test as well, specifically to simulate an event receiver, such as a tool or stationary system receiving device or product events.

Run **ubisense_open_protocol_clientserver(.exe)** at the command prompt in a shell or a Command Prompt window without any parameters to get the list of available options:

```
ubisense_open_protocol_clientserver.exe
Usage: ubisense_open_protocol_clientserver.exe server <port> <testcase>
   or: ubisense_open_protocol_clientserver.exe server <server> <port> <testcase>
   or: ubisense_open_protocol_clientserver.exe client <server> <port> <testcase>
   or: ubisense_open_protocol_clientserver.exe server <server> <port> 1
<CommunicationStartMinimumRevision>
Where testcase
  1: Connect, then listen
  2: Connect, then listen, but dont answer
  3: Connect, Disconnect
  4: Connect, Send tool tag ID
  5: Connect, then listen, respond with error code
  6: Connect, then listen, respond with delay
100: Connect, Send messages from file
```

The test case which is useful for application testing is

```
  1: Connect, then listen
```

Event receiver simulation

Use test case 1 to simulate an external system receiving event information from ACS, such as a tool controller or a PLC. The simulation tool will connect to ACS, receive event telegrams, such as MID0042, MID0043, and MID0150 and respond with necessary response telegrams.

This behavior can be simulated with test case 1.

```
ubisense_open_protocol_clientserver.exe [client|server] <server> <port>
```

For example:

```
ubisense_open_protocol_clientserver.exe server 127.0.0.1 4045 1
```

ubisense_open_protocol_clientserver

In this setup the simulator would

- act as a TCP server
- allow ACS to connect on IP address 127.0.0.1 and port 4045
- run with testcase 1

There are no specific parameters for test case 1.