



THIS SERVICE MANUAL IS ISSUED IN ADOBE ACROBAT FORMAT FOR EASE OF VIEWING AND DISTRIBUTING AND CAN BE VIEWED AND PRINTED AS REQUIRED.

A LIST OF CHAPTERS IS SHOWN ON THE LEFT HAND SIDE OF THE SCREEN. CLICKING ON AN ARROW WILL DISPLAY THE CONTENTS OF THE CHAPTER. TO VIEW THE REQUIRED CHAPTER OR SECTION, CLICK ON THE TEXT OF THE REQUIRED HEADING.

THE LIST CAN BE HIDDEN BY CLICKING THE LEFT ICON ON THE TOOLBAR AND WILL BE DISPLAYED BY CLICKING ON THE SECOND TO LEFT ICON.

TO MOVE THROUGH THE MANUAL, USE THE SCROLLBAR OR THE SINGLE ARROWS IN THE TOOLBAR (PAGE UP AND PAGE DOWN). THE DOUBLE ARROWS ON THE TOOLBAR MOVE TO THE FIRST AND LAST PAGES.

THE PAGE CAN BE ZOOMED IN BY CLICKING ON THE MAGNIFYING GLASS IN THE TOOLBAR. THIS TOOL WILL ALSO ZOOM OUT (e.g. WHEN VIEWING CIRCUIT DIAGRAMS) IF THE CTRL KEY IS PRESSED WHEN CLICKING ON THE PAGE.

THE PAGE WILL BE EXPANDED IF THE RIGHT PAGE LAYOUT ICON IS CLICKED JUST TO THE LEFT OF THE SEARCH BINOCULARS ICON. THE FULL PAGE WILL BE DISPLAYED IF THE MIDDLE PAGE ICON IS CLICKED.

SOME OF THE TEXT IN THE MANUAL IS COLOURED:- CLICKING ON BLUE TEXT WILL MOVE TO A LOCATION IN THE SAME CHAPTER AND CLICKING ON RED TEXT WILL MOVE TO A LOCATION IN A DIFFERENT CHAPTER.

TO RETURN TO THE ORIGINAL POSITION AFTER MOVING THROUGH A LINK, CLICK THE RIGHT MOUSE BUTTON THEN SELECT GO BACK.

THIS PROCEDURE WILL ALWAYS MOVE BACK THROUGH THE SCREENS EVEN THOUGH A LINK WAS NOT USED.

LLOYD INSTRUMENTS LTD

APPLICATION MANUAL

FOR NEXYGEN ONDIO

VERSION V2.0

ISSUE 1.0 - NOVEMBER 1998

PART NUMBER 01/2871

LLOYD INSTRUMENTS LTD

12 Barnes Wallis Road

Segensworth East

Fareham

Hampshire

PO15 5TT

Telephone: +44 (0) 1489 574221

Fax: +44 (0) 1489 885118

ISSUED BY CUSTOMER SUPPORT DEPT

Direct phone line +44 (0) 1489 574226



W.D. Turner Company, Inc.

Sales & Technical Services

#7 Whitaker Place
Thomasville, North Carolina 27360

Phone: (336) 882-4931

Fax: (336) 882-5175

www.WDturner.com

NOTICE

This Manual or Spec Sheet was provided through our web site to assist Customers who purchased equipment through W. D. Turner Company, Inc.

Please do not operate, repair or modify equipment without proper training. Our company provides on-site calibrations, repairs and training. Please contact our office for free quotes or more information.

Please use this Manual or Spec Sheet at your own risk. We welcome inquires about this equipment. Telephone and email technical support is provided free for our customers.

You are invited to visit our web site at:

www.wdturner.com

Thank you for considering our products and services!

TABLE OF CONTENTS

1	INTRODUCTION	
1.1	General	1
1.2	Differences between an Ondio Setup and a Predefined Nexygen Setup	1
1.3	Differences between an Ondio Setup and an R Control Setup	1
1.4	Installing Ondio	2
1.5	Selecting an Ondio Test Setup	2
1.6	Creating an Ondio Test Setup	3
2	DEFINING AN ONDIO TEST	
2.1	General	1
2.2	Pre and Post Test Questions	1
2.3	Machine Direction	2
2.4	Preload	2
2.5	Sample Height and Automatic Height Measurement	3
2.6	Sample Cross Sectional Area	4
2.7	Break Detector	5
2.8	Auto Zero and Auto Return	5
2.9	Post Test Markers	6
2.10	Primary and Secondary Scripts	6
3	PROGRAMMING PRINCIPLES	
3.1	Defining a Test	1
3.2	Typing the Script	1
3.3	Sending Commands to the Materials Testing Machine	2
3.4	Measuring Results	3
3.5	Primary Script Commands	4
3.6	Test Results	5
3.7	Sample and Test Information	6
3.8	Outputting Data and Messages	6
3.9	Programming Notes for VB Script	6
3.10	Programming Notes for STAGE and STAGEHOLD	10
3.11	Programming Notes for RESULT	11
3.12	Programming Notes for STATIC and STAGESTATIC	12
3.13	Programming Notes for PROGRESS	12
3.14	Programming Notes for MODULUS and OFFSET YIELD	13
3.15	Programming Notes for BREAK Results	14
3.16	Programming Notes for BREAKON and BREAKOFF	14
3.17	Programming Notes for ADVANCEON and ADVANCEOFF	15
3.18	Programming Notes for GLOBAL	17
3.19	Programming Notes for User Variables	18
3.20	Tables of Valid Units	20
4	EXAMPLE SCRIPTS	

1. INTRODUCTION

1.1 General

Ondio V2.0 is a fully customisable Test Setup for use with NEXYGEN V2.0 and is only required if one of the predefined Test Setups supplied with NEXYGEN does not perform a test exactly as required. The predefined test setup's are written to conform to recognised standards but sometimes these are not suitable because additional results are required or a more specialised test is to be performed. Ondio is supplied as a separate product so uses the NEXYGEN features and benefits such as Break Detector, Extra Results etc. The testing machine is controlled and the required results measured by writing an Ondio Test setup using Microsoft VB Script using a concept similar to the Lloyd Instruments Windows R Control.

Typical uses for an Ondio setup are:-

- 1 To control the machine in a specialised way, e.g. cycle 10 times then apply a constant load.
- 2 To obtain specialised results that are not automatically calculated by one of the predefined Test Setups, e.g. Secant Modulus.
- 3 When user wants to perform a test which does not exactly conform to a recognised standard.

1.2 Differences between an Ondio Setup and a Predefined Nexygen Setup

- 1 The Break Detector is used to indicate that the load value has dropped and does not automatically end the test.
- 2 The test can be controlled by load drop, the advance button and the end button as well as the usual measured values of load, extension etc.
- 3 Markers can be selected for post test analysis

1.3 Differences between an Ondio Setup and an R Control Setup

- 1 The Break Detector is used to indicate that the load value has dropped and does not automatically end the test.
- 2 The test can be controlled by load drop, the advance button and the end button as well as the usual measured values of load, extension etc.
- 3 The Relative and Absolute commands are no longer required when the test is to be controlled by a drop in load.
- 4 More versatile programming using VB script instead of individual action lines

- 5 Unlimited number of stages instead of the maximum of 30 stages
- 6 Improved syntax checking of script
- 7 Script can be cut/pasted from examples, e-mail etc
- 8 Use of comments in script
- 9 Easier cycling setup's using a FOR NEXT loop
- 10 Unlimited number of results so can report max load for say 500 cycles
- 11 Unlimited number of markers instead of the maximum of 8 cursor points
- 12 Easier reporting of results using single result function in primary script. Only need to remember variable names when using the secondary script.

1.4 Installing Ondio

- 1 Ondio can only be operated from within NEXYGEN so this must be installed first as shown in the NEXYGEN Training Manual.
- 2 Insert the Ondio CD into the CD drive.
- 3 When the Wizard starts, select the required options and directories etc. as requested.
- 4 The Setup program may update the Windows 95 operating system to allow VB Script to run and will also copy the VB Script and Ondio files.

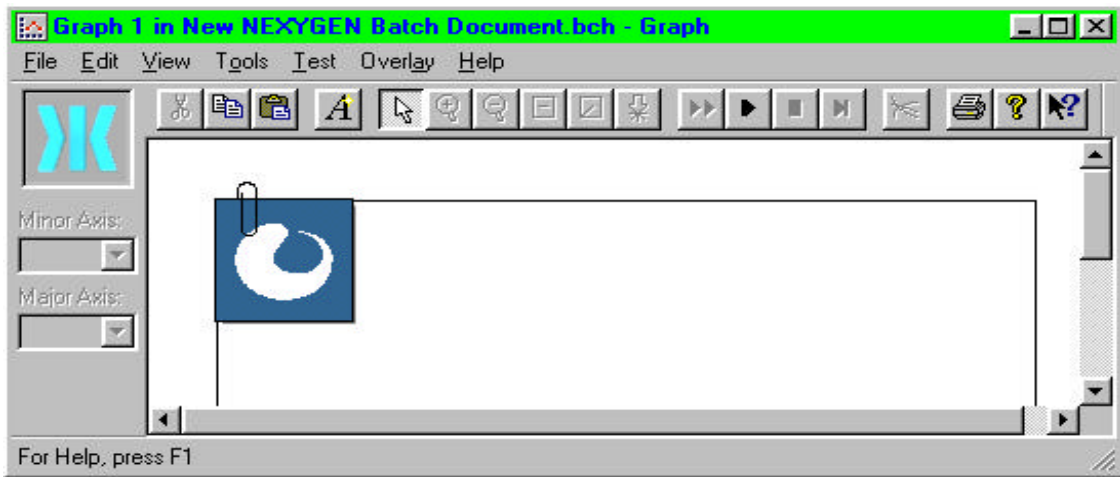
1.5 Selecting an Ondio Test Setup

An Ondio batch can be created in exactly the same way as any other Nexygen batch as shown below:-

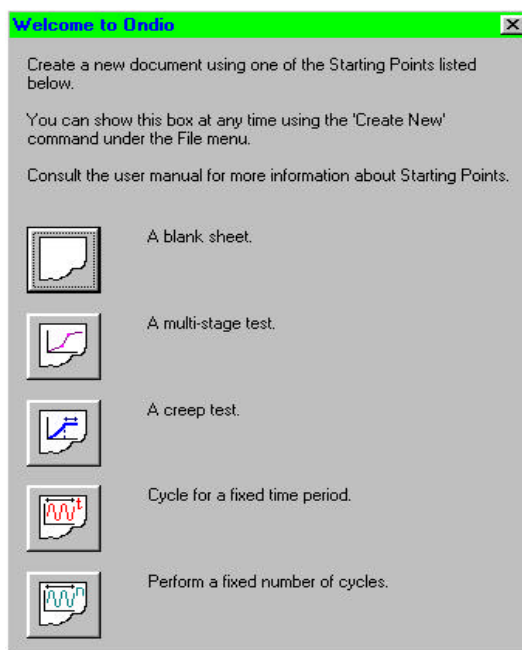
- 1 **RIGHT CLICK** anywhere on the desktop.
- 2 Select NEW, NEXYGEN BATCH DOCUMENT.
- 3 **DOUBLE CLICK** on the required UNCATEGORISED Category.
- 4 **DOUBLE CLICK** on ONDIO DOCUMENT.
- 5 Type a suitable name for the Icon, e.g. **5 Cycle**, noting that spaces are allowed (longfilename facility) but characters such as /, \, ? and * are not allowed.
- 6 Click on FINISH to create the Icon.

1.6 Creating an Ondio Test Setup

Start the batch **DOUBLE CLICKING** on the Icon to display the batch table. A batch containing an Ondio **TEST SETUP** behaves exactly the same way as any other NEXYGEN batch so the columns can be hidden or re-ordered as required. Click on the Start test Icon, or press the **F5** key, to display the graph window, then the graph can be resized as required. The graph screen contains a paper-clipped Ondio **TEST SETUP** as shown below:-



The Ondio **TEST SETUP** does not display any test information directly so has to be activated before any parameters can be defined or altered. To define the test, **DOUBLE CLICK** on the **TEST SETUP** to activate the Ondio Setup Editor. The first time the Setup Editor is activated in a new batch, the Setup Wizard shown below will offer 4 basic examples which may be modified to suit the required test. These 4 examples show a multi-stage test, a creep (constant load) test and two cycle tests. There is also a blank sheet which is used when no predefined example is required.



After selecting one of the 5 options, the Test Parameters screen of the setup editor will be displayed as shown in the next section.

2. DEFINING AN ONDIO TEST

2.1 General

The top part of the screen shows both a menu bar and a toolbar, both of which are visible when the program is first installed. The menu bar has 4 options and is always visible. The toolbar contains 7 of the main functions and this may be hidden if not required. The bottom part of the table shows three tabs to switch between the Test Parameters, Primary Script or the Secondary Script pages. The bottom part of the screen shows a status bar which may be hidden if not required. The FILE MENU and TOOLBAR functions are similar to the NEXYGEN functions so only the new features are described here.

The FILE menu contains an option CREATE NEW which will display the Setup Wizard which allows a completely different type of test to be defined during a series of tests. The EDIT menu contains options of PRE-TEST EXTRA RESULTS and POST-TEST EXTRA RESULTS which are described later. These results can also be accessed by the TOOLBAR Icons showing notepads with the numbers 1 (Pre-Test) and 2 (Post-Test). The VIEW menu contains options of displaying the PARAMETERS, PRIMARY SCRIPT or SECONDARY SCRIPT and these can also be accessed by the Tabs at the bottom of the Ondio window.

The main part of the Test Parameters screen is used to define the positive direction of the graph, preload value, sample height, sample cross sectional area, break detector, Auto Zero, Auto Return and the number of post test markers that may be required.

2.2 Pre and Post Test Questions

These screens allow pre-test and post-test questions to be modified or new ones to be added. To change the text displayed for a question, RIGHT CLICK on the title, select RENAME, type the required title then press the ENTER key. To add a new question, click on ADD RESULT, select the type of data to be entered, type the name then press the ENTER key. When all questions are correct, click on OK to return to the Ondio Setup Editor.

The types of data that can be entered are:-

- Text - Normal Questions
- Yes/No - Questions that are answered by a tick box for Yes
- Numerical - Numerical Questions that can include units for Auto Conversion

When the question names and data type are correct, the properties can be modified to suit the test. When a test is started, a Text question can either display the last entry or a list of the last 5 entries. To select the required action, click onto the title of text question then click on the properties button. When a test is started, a Numerical question will request either a value of a value with units, e.g. kg and will only accept values within a defined range. To select the units and range, click onto the title of Numerical question then click on the properties button.

2.3 Machine Direction

Clicking on the **DIRECTION** drop down arrow provides a choice of Tension or Compression. If TENSION is selected, the limit will normally define a tensile load and a position above zero unless a negative limit is used when it will define a compressive load or a position below zero. If COMPRESSION is selected, the limit will normally define a compressive load and a position below zero unless a negative limit is used when it will define a tensile load or a position above zero. Note that the graph can draw both positive and negative loads and extensions so a Cycle Through Zero option is not required.

2.4 Preload

Clicking on the **PRELOAD** selector (+) displays the screen shown below:-



The preload function rezeros the extension after the preload force has been applied to eliminate any crosshead movement that occurs when the sample is not loaded. This is suitable for compression or bending tests where the initial grip separation is more than the height of the sample. The graph extension axis will be labelled "**Machine Extension**" and the crosshead will initially move at the preload speed until the preload force is applied.

Once the preload has been applied, a new axis called "**Extension**" will be created with zero extension at the preload force. The main part of the test is then performed using the actions specified in the Primary Script. If the preload function is switched off, the "machine Extension" axis will **NOT** be created.

Note that if the preload function is used, ALL extension stage limits are referenced to the Pre-Load position and NOT to the start of test position, i.e. a limit of 0mm will move the crosshead to the preload position.

To return the crosshead to the start of test position, e.g. to allow the sample to expand after during a compression test, use a stage limit of say -5mm to give a 5mm clearance between the sample and the compression plates.

Note that the crosshead WILL return to the start of test position either during an Automatic Return or if the Return button is clicked.

2.5 Sample Height and Automatic Height Measurement

Clicking on the **HEIGHT** selector displays the screen shown below which provides four different options regarding the height of the sample.

I am not interested in height
 Samples are auto-measured against a datum position
 You should manually zero the machine before measuring the datum position
 Preload:
 Speed:
 I know the height of the samples

 Operator can 'fine tune' the height

The first option ignores the sample height so the crosshead movement can only be reported as an extension. However, if a strain result is required, the sample height is required and can be entered using the other 3 options. The second option enables the automatic height measuring system which requires a datum position to be measured. The third option allows the nominal height to be manually entered on this screen and the fourth option allows the user to modify the nominal height before each sample is tested.

The automatic height measurement feature is used as follows:-

- 1 Select the compression direction.
- 2 Select the PRELOAD feature and specify the PRELOAD (CONTACT) FORCE and SPEED required for the test.
- 3 Select the Height Auto Measure feature and specify the required DATUM FORCE and SPEED. Some tests require the Datum force to be the same as the test maximum force.
- 4 Ensure that the **AUTO ZERO** feature is switched **OFF**.
- 5 Attach the required grips, e.g. compression plates, move the crosshead using the jog keys on either the machines console or the software console to give the required grip separation then **ZERO** the machine.
- 6 Do **NOT** click on the **START TEST** Icon or press the **F5** key to display the pretest dialogue box. If this has been displayed by accident, click on the **ABORT** button to close it and return to the graph screen.
- 7 Manually move the crosshead down, using the jog keys on either the machines console or the software console, until the grip separation is approx. 5mm. **DO NOT PRESS THE ZERO BUTTON ON THE MACHINE OR THE SOFTWARE CONSOLE.**

- 8 Click on the **START TEST** Icon or press the **F5** key to display the pretest dialogue box. Enter the required details then click on the OK button. The crosshead will move downwards at the DATUM SPEED to apply the DATUM force and this position is used as the reference DATUM position.
- 9 The crosshead will automatically return at maximum speed to the position set in step 5.
- 10 Fit the sample then click on the OK button to start the test.
- 11 The crosshead will move downwards at the PRELOAD SPEED until the PRELOAD force is applied. The TRUE HEIGHT of the sample is the distance between the crosshead position and the datum position and this measurement is automatically stored in the **HEIGHT** variable and can be reported as a Result or used to define a Stage limit.
- 12 The Primary Script will now control the machine and the test will be performed.
- 13 **THE ZERO BUTTON MUST NOT BE PRESSED DURING THE TESTING.**

2.6 Sample Cross Sectional Area

Clicking on the **AREA** selector displays the screen shown below which provides three different options regarding the cross-sectional area of the sample.

I am not interested in cross sectional area
 I know the width and breadth of the samples
 I know the diameter of the samples
 I know the cross sectional area of the samples

6.000 mm
 3.000 mm
 1.354 mm
 3.000 mm²

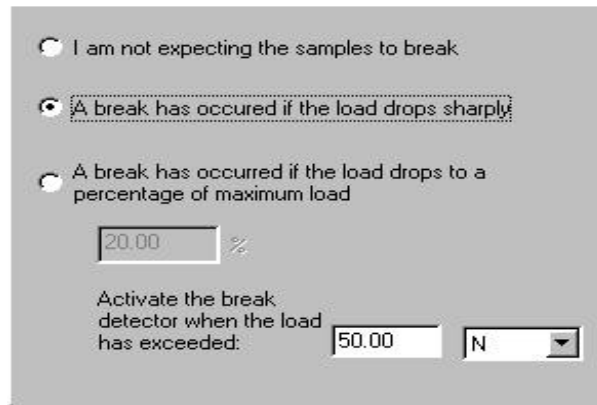
Operator can 'fine tune' the area

The first option ignores the sample area so the force applied to the sample can only be reported as a load. However, if a stress result is required, the sample dimensions are required and can be entered using the other 2 options. The second option allows the nominal sample dimensions to be manually entered on this screen and the third option allows the user to modify the nominal dimensions before each sample is tested.

The cross sectional area is automatically calculated for rectangular or circular cross sections, but has to be manually calculated for other sections, e.g. hexagonal etc.

2.7 Break Detector

Clicking on the **BREAK** selector displays the screen shown below which provides three different options regarding the cross-sectional area of the sample.



The first option disables the break detector so the test will not respond to a sample break. The second option will detect a break if the load falls quickly so is suitable for samples that break quickly, **usually with some noise**. The third option will detect a break if the load falls to a percentage of the last measured maximum load so is suitable for **samples that crack or tear** so that the load falls too slowly for the fast Break Detector to operate correctly. Therefore this is mainly used as a safety feature so it is important to note that in this case the Load at Break value will **NOT** give a useful result. The Break Detector is inhibited at low load values to prevent it from triggering if the load falls slightly due to grip movement or noise etc. and the threshold load is entered in the activate box.

The Break Detector can cause a test to end earlier than expected if the load falls during the test. A common reason for the load to fall is when a cyclic test is being performed and the test will end when the load falls close to zero during the first cycle. Therefore, if the Break Detector is to end a cyclic test when the sample breaks, it **MUST** be switched on and off during the test by using the BREAKON and BREAKOFF commands.

Note that although the Break Detector is commonly used to end the test at sample break, it can also be used to end the current stage when the load drops. Also note that the Break Detector detects a break when the load falls back towards zero so operates in both Tension and Compression modes. These features can be used to control a specialised test, e.g. push a cap onto the top of a bottle to find the "Push On Force" then pull the cap off the bottle to find the "Pull Off Force". In this case, the break detector **MUST** be switched off when the initial load falls to zero so 3 stages are required, push on with break detector active, return to zero load with break detector off and pull off with break detector active.

2.8 Auto Zero and Auto Return

Clicking on the **AUTO ZERO** Check Box turns the Auto Zero feature On and Off. When Auto Zero is selected, the machine Load and Extension will be set to Zero before every test. However, this feature can cause incorrect load readings especially if wedge grips are being used.

When a wedge grip is tightened, a compressive force may be applied to the sample and if this is zeroed out by the Auto Zero function, a load offset is produced and any measured load values will be higher than expected. **It is therefore recommended that the machine is Zeroed before a BATCH of tests is started and NOT zeroed between each test.**

Clicking on the **AUTO RETURN** Check Box turns the Auto Return feature On and Off. When Auto Return is selected, the crosshead will return to it's home position at the end of every test. Note that the home position is the position where the crosshead was when the machine was last rezeroed so is usually the start of test position.

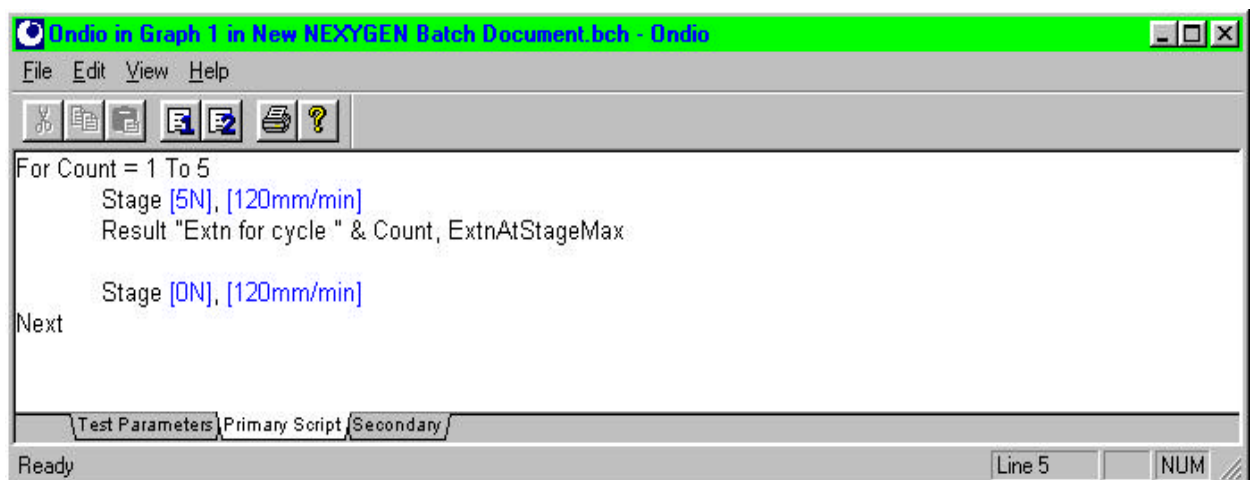
Note that the use of Automatic Return may cause damage to the loadcell and/or the grips if the crosshead returns after breaking a stiff sample, e.g. a metal sample. This is because the sample will generally have increased in length during the test so a compressive force will be applied to the loadcell and grips as the crosshead returns at maximum speed to it's home position.

2.9 Post Test Markers

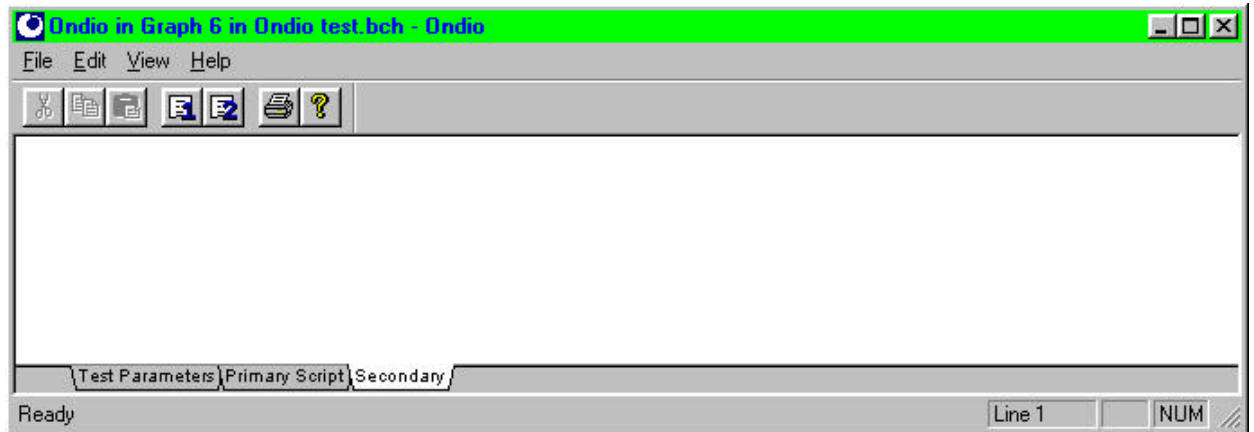
Entering a number in the **MARKERS** field will place the selected number of markers on the graph, at equal time intervals, when the test has finished. These markers can be moved to their required positions after the test by clicking on the "Move Marker" Icon on the toolbar then the load, extension or work at the marker points can be obtained by marker variables.

2.10 Primary and Secondary Scripts

When the setup parameters have been selected, click on the Primary Script tab to display the primary script editing screen which is read during the test so controls the machine and obtains results in real time. The test action has to be defines using VB Script and the example below shows the predefined script entered by the setup wizard for the "Perform a fixed number of cycles" test. The tests supplied by the wizard may be easily modified or completely new tests can be defined by selecting the blank sheet option.



When the primary script has been typed, click on the Secondary tab to display the secondary script editing screen. This screen is optional and is not used in the "Perform a fixed number of cycles" example. This script is only read after the test has finished so any results that may require modification or recalculation post test should be entered here.



The following sections describe the programming concepts and the editor screens in detail.

3. PROGRAMMING PRINCIPLES

3.1 Defining a Test

Before an Ondio test setup can be written, the test has to be thought of as a series of actions which control the movement of the machine and obtain the test results that are to be measured in **real time, i.e. during the test**. The machine can be made to move in one direction at one speed, change speed, change direction or can change operating mode from constant speed to creep mode, relaxation mode or load rate control. When the sequence of actions has been determined, the script has to be written to perform these actions. The script is generally actioned line-by-line but this can be modified by using VB Script statements such as **IF - THEN - ELSE**, **FOR - NEXT** or **WHILE - WEND**. The script is always actioned but the commands to move the machine can be disabled, if required, by the Break Detector, the ADVANCE button or the END button. Note that there are no LABEL or GOTO commands so the script has to be written so that the required actions occur sequentially even within an IF - THEN - ELSE statement. The Ondio script will consist of standard VB script commands together with Ondio commands. For example, if a cycle test is required, 2 stages are used, one to go up and one to go down. These stages do not obtain any measurements so if the load at the upper limit is required, an additional line of script is used as shown in the example below:-

```
GO UP
MEASURE LOAD
GO DOWN
```

If the same test is to cycle 50 times, then a FOR - NEXT loop is used as shown below:-

```
FOR CYCLE = 1 to 50
    GO UP
    MEASURE LOAD
    GO DOWN
NEXT
```

3.2 Typing the Script

When the sequence of actions and results has been determined, the script is typed in the primary script screen. The text will be displayed in **black** as it is typed but will change colour according to its content. If any value that contains a unit is required, e.g. 500N or 100mm/min, these **MUST** be enclosed in square brackets []. As soon as the opening bracket is typed, the remaining text will be shown in **red** to show that the unit has not been entered correctly. When the closing bracket is typed, the text will change to **blue** providing that a valid unit has been entered. Therefore, [5N] will be shown in blue but [5n] will be shown in red because the abbreviation for Newtons is a capital N. Any **CALCULATION** containing units **MUST** be **FULLY ENCLOSED** in square brackets [] and will be displayed in **purple** when the formula has been entered correctly. If any comments or notes are required in the script to aid the understanding of the test, these **MUST** be preceded by an apostrophe ' and as soon as this is typed, the remaining text will be displayed in **green** and will not be actioned by the program.

3.3 Sending Commands to the Materials Testing Machine

To move the machine, a **STAGE** command can be used and this command requires two additional values of a stage **limit** and the required crosshead **speed**. When this line of script is actioned, the machine will immediately move at the specified speed until the limit condition is reached. The speed can be a fixed speed (e.g. mm/min) or a load rate (e.g. N/min). If the required limit is a load, stress or work, the limit value may be either an absolute value or a value relative to one previously measured. If the required limit is an extension, the crosshead can move to a position which is either an actual position or a position dependant upon a previously measured extension.

When the limit condition has been reached, the values of load, extension, time and work are automatically measured and stored into Ondio variables. The next line of the script is then actioned and will usually either store this data or move the machine to the next required position. If any measured values are required as a result, they **MUST** either be output to the batch table using the **RESULT** command or stored into a user defined variable using the **SET** command which stores a **value containing a unit**.

If the test consists of more than one stage, the machine will pause for a short period between stages because the next stage will **NOT** be sent to the machine until the previous stage has been completed. If it is important to have minimal delay, the information for the next stage has to be sent to the machine **BEFORE** the current stage is started. This is achieved by replacing the **STAGE** commands with **SETUPSTAGE** commands which inform the machine of the required stages. However, these stages are not performed until a **RUNSTAGE** or **RUNALLSTAGES** command is actioned by the script. The **RUNSTAGE** command will run the next unactioned **STAGE** that has been sent to the machine and the **RUNALLSTAGES** command will run all unactioned **STAGES** sequentially.

A two stage test without real time measurements could be written in three ways:-

1	STAGE STAGE	Delay
2	SETUPSTAGE SETUPSTAGE RUNSTAGE RUNSTAGE	No delay
3	SETUPSTAGE SETUPSTAGE RUNALLSTAGES	No delay

A two stage test with real time measurements could be written in two ways:-

1	STAGE RESULT STAGE	Delay
2	SETUPSTAGE SETUPSTAGE RUNSTAGE RESULT RUNSTAGE	No delay

The first RUNSTAGE command moves the machine to the limit defined by the first SETUPSTAGE, the RESULT is stored in the batch table then the second RUNSTAGE command moves the machine to the limit defined by the second SETUPSTAGE. Note that in this case, the RUNALLSTAGES command cannot be used because the required result could not be measured.

To set the machine into either a constant load mode (creep mode) or a relaxation mode, a **STAGEHOLD** command can be used and this command requires three additional values of a stage **limit**, initial crosshead **speed** and the required hold **time**. When this line of script is actioned, the machine will initially move at the specified speed until the specified limit condition is achieved then will maintain this condition for the specified hold time. If the limit is a load, the machine will exert a constant load on the sample for the time period. If the limit is an extension, the crosshead will stay at this extension for the time period.

At the END of the hold time, the values of load, extension, time and work will be automatically measured then the next line of script will be actioned. If any of measured values are required as a result, they MUST either be output to the batch table using the **RESULT** command or stored into a user defined variable using the **SET** command.

3.4 Measuring Results

At the **END** of EVERY STAGE, even if the stage was ended early by clicking on the Advance button or End button (when activated), the values of Load, Extension, Time and Work are automatically stored in the variables **LOAD**, **EXTN**, **TESTTIME** and **WORK**. The values of Load, Extension and Work will also be measured if the stage is ended early by the Break Detector triggering, but these values should **ONLY** be used as an indication that the load dropped, e.g. for a snap on type of test and should **NOT** be used as break results at a true sample break. The true break values are obtained from secondary screen variables.

Additional results are also available for each STAGE and these are the Maximum Load, Extension at Maximum Load, Minimum Load, Extension at Minimum Load and any Stage Static Loads or Extensions. If any of these results are required they MUST either be output to the batch table using the **RESULT** command or stored into a user defined variable using the **SET** command in the **Primary Script**.

3.5 Primary Script Commands

Command	Function
Stage	Syntax - Stage [Limit], [Speed] Sends a limit stage to machine then performs the stage
SetupStage	Syntax - SetupStage [Limit], [Speed] Sends a limit stage to machine but this is not performed until RunStage
RunStage	Performs next stage previously sent to machine
RunAllStages	Performs all stages previously sent to machine
StageHold	Syntax - StageHold [Limit], [Speed], [Hold Time] Sends a hold stage to machine then performs the stage
SetupStageHold	Syntax - StageHold [Limit], [Speed], [Hold Time] Sends a hold stage to machine but this is not performed until a RunStage
BreakOff	Inhibits the Break Detector and resets the variable " Broken "
BreakOn	Enables the Break Detector after being inhibited by a BreakOff command and resets the variable " Broken ". Note that the Break Detector must be selected on the Test Parameters screen
Broken	A Boolean Variable that indicates whether the Break Detector has been triggered. When set to True , the variables Load, Extn, Time and Work will contain the values at the " Break Point ". All Stage and RunStage commands are ignored until " Broken " is reset.
AdvanceOff	Disables the Advance button and resets the variable " Advancing "
AdvanceOn	Enables the Advance button and resets the variable " Advancing "
Advancing	A Boolean Variable that indicates whether the Advance button has been clicked since the last AdvanceOn command. When set to True , the variables Load, Extn, Time and Work will contain the values when the button was pressed. All Stage and RunStage commands are ignored until " Advancing " is reset
EndOff	Disables the End button and resets the variable " Ending "
EndOn	Enables the End button and resets the variable " Ending "
Ending	A Boolean Variable that indicates whether the End button has been clicked since the last EndOn command. When set to True , the variables Load, Extn, Time and Work will contain the values when the button was pressed. All Stage and RunStage commands are ignored until " Ending " is reset
TearOn	Selects the Tear Test Mode with no graph smoothing
TearOff	Selects Normal Test Mode with some graph smoothing
StopMachine	Stops the machine and disables the motor drive until the next STAGE or RUNSTAGE command

3.6 Test Results

Variable	Function	Primary	Secondary
Load	Load at END of previous stage	Yes	
Extn	Extension at END of previous stage	Yes	
TestTime	Time at END of previous stage	Yes	
Work	Work from preload to END of previous stage	Yes	Yes
LoadAtStageMax	Maximum Load during previous stage	Yes	
ExtnAtStageMax	Extension at Maximum Load of previous stage	Yes	
LoadAtStageMin	Minimum Load during previous stage	Yes	
ExtnAtStageMin	Extension at Minimum Load of previous stage	Yes	
StageStaticLoad	Syntax - StageStaticLoad(extension) Load at the FIRST occurrence of the extension during the previous stage	Yes	
StageStaticExtn	Syntax - StageStaticExtn(load) Extension at the FIRST occurrence of the load during the previous stage	Yes	
LoadAtMax	Maximum Load of the whole test	Yes	Yes
ExtnAtMax	Extension at Maximum Load of the whole test	Yes	Yes
LoadAtMin	Minimum Load of the whole test	Yes	Yes
ExtnAtMin	Extension at Minimum Load of the whole test	Yes	Yes
LoadAtBreak	Load at Break - No break gives ERROR		Yes
ExtnAtBreak	Extension at Break - No break gives ERROR		Yes
WorkAtBreak	Work from preload to Break No break gives an ERROR		Yes
StaticLoad	Syntax - StaticLoad(extension) Load at the FIRST occurrence of the extension	Yes	Yes
StaticExtn	Syntax - StaticExtn(load) Extension at the FIRST occurrence of the load	Yes	Yes
Modulus	Tangential Modulus in Load/Extension Units, NOT MOE. * See note below.		Yes
LoadAtOffsetYield	Load at point defined by OffsetExtn. * See note below		Yes
ExtnAtOffsetYield	Extension at point defined by OffsetExtn. * See note below.		Yes
SetOffsetExtn	Syntax - SetOffsetExtn extension Extension to define Yield Point		Yes
MarkerLoad	Syntax - MarkerLoad(number) Load at previously selected Marker indicated in brackets.		Yes
MarkerExtn	Syntax - MarkerExtn(number) Extension at Marker indicated in brackets		Yes
MarkerWork	Syntax - MarkerWork(number) Work from preload to Marker		Yes

* **Note:-** When any of these results are selected, the steepest slope will be marked on the graph together with the tangential line

3.7 Sample and Test Information

Variable	Function	Primary	Secondary
Height	Sample Height or Gauge Length	Yes	Yes
Width	Sample Width	Yes	Yes
Breadth	Sample Breadth	Yes	Yes
Diameter	Sample Diameter	Yes	Yes
Area	Sample Cross Sectional Area	Yes	Yes
StageNumber	Number of stages performed	Yes	
Broken	Indicates whether the Break Detector has been triggered. When set to True , the variables Load, Extn, Time and Work will contain the values at the " Break Point ".	Yes	Yes

3.8 Outputting Data and Messages

Command	Function
Result	Syntax - Result "Batch Column Title", Result Stores the Result into the column labelled " Batch Column Title " in the batch table, automatically creating this column if it doesn't exist
Progress	Syntax - Progress "Message" Displays the message on the Status Bar during the test
Global	Syntax - Global "Name", Value Stores a value measured in the Primary Script for use in the Secondary Script
Global	Syntax - Global ("Name") Retrieves a value in the Secondary Script

3.9 Programming Notes for VB Script

IF - THEN - ELSE

An **If - Then - Else** statement is used to determine which lines of the script will be actioned depending upon a measured condition. This could be used to select different stage limit depending upon a measured load value as shown below:-

```

STAGE [10mm], [100mm/min]

IF [Load < [100N] ] THEN
    STAGE [25mm], [100mm/min]
    RESULT "Load at 25mm", Load
ELSE
    STAGE [50mm], [100mm/min]
    RESULT "Load at 50mm", Load
END IF
    
```

The sample is stretched by 10mm and the load measured. If the load is less than 100N then the sample is stretched by 25mm but if the load is more than 100N then the sample is stretched by 50mm. In both cases, the load at this stretch is measured. Note that the comparison value of **100N** is a value **WITH UNITS** so **MUST** be enclosed by square brackets []. Also note that the formula of Load < [100N] is a calculation **WITH UNITS** so must be **FULLY ENCLOSED** by square brackets.

FOR - NEXT

A **For - Next** statement is used to repeat certain lines of script for a fixed number of times, so can be used to create a cyclic test. The For - Next statement uses a variable which is automatically incremented from the starting value to the ending value. The lines of script between the For statement and the Next statement are repeated until the variable contains the ending value when the script will continue after the Next statement. A simple 10 cycle test is shown below:-

```
FOR Cycle = 1 to 10
    STAGE [25mm], [100mm/min]
    STAGE [0mm], [100mm/min]
NEXT
```

Here the value of the variable called **Cycle** starts at 1 and ends at 10 so the stages are repeated 10 times to give a 10 cycle test between 25mm and 0mm.

EXIT FOR

A **For - Next** loop may be exited before the variable contains the ending value by an EXIT FOR command, usually together with an IF - THEN - ELSE statement as shown below:-

```
FOR Cycle = 1 to 1000
    STAGE [25mm], [100mm/min]
    IF [Load < [100N] ] THEN EXIT FOR
    STAGE [0mm], [100mm/min]
NEXT
```

```
IF cycle < 1000 THEN RESULT "Stop Load", Load
```

Here the value of the variable called **Cycle** starts at 1 and ends at 1000 so the stages are repeated 1000 times to give a 1000 cycle test between 25mm and 0mm. However, the load is measured at 25mm on every cycle and if the measured load falls below 100N, then the test will end before all 1000 cycles have been performed. Note that the comparison value of **100N** is a value **WITH UNITS** so **MUST** be enclosed by square brackets []. Also note that the formula of Load < [100N] is a calculation **WITH UNITS** so must be **FULLY ENCLOSED** by square brackets. The last line will report the load value which caused the test to end if all 1000 cycles were not performed. Note that the comparison value of 1000 is a value WITHOUT UNITS so does not need square brackets either around the 1000 or around the calculation formula.

WHILE - WEND

A **While - Wend** statement is used to repeat certain lines of script depending upon a measured condition so can be used to create a cyclic test where the number of cycles is not fixed but is to run for a say a time period. The While - Wend statement uses a variable which is checked at the beginning of the While statement and if the condition is TRUE, the lines of script between the While statement and the Wend statement are repeated. If the condition is FALSE, the script will continue after the Wend statement. The example below will cycle for 10 minutes.

```
WHILE [TestTime < [10min] ]  
    STAGE [25mm], [100mm/min]  
    STAGE [0mm], [100mm/min]  
WEND
```

The value of TestTime will be checked and if it is less than 10 minutes, the two stages will be performed. At the end of the second stage, the TestTime will be stored and checked by the while statement. If the TestTime is still less than 10 minutes the stages will be repeated. The test will continue until the value of TestTime at the **END** of the **SECOND** stage is more than 10 minutes then the test will end. The comparison value of 10min is a value **WITH UNITS** so **MUST** be enclosed by square brackets [] and the complete formula of **TestTime < [10min]** is a calculation **WITH UNITS** so must ALSO be **FULLY ENCLOSED** by square brackets.

Note that a formula of TestTime = [10min] will **NOT** perform correctly because the test will only stop if the TestTime at the end of the second stage is **EXACTLY** 10.000 minutes! The same principle applies to ANY measured parameter unless it will have a definite value, e.g. a cycle counter. Also note that the WHILE - WEND loop is not used in any of the examples shown in section 4 because generally the number of cycles performed is required so it is easier to use a FOR - NEXT loop with an EXIT FOR command.

Storing Values into Variables

The measured results are usually entered directly into the batch table using the Ondio's RESULT command. However, these results can be stored into user defined variables for use in stage limits etc. To store a value **WITH UNITS**, e.g. measured Load etc. use the **SET** command to copy the value from one variable to another together with an **equals sign**, e.g.

```
SET Lastload = Load
```

To store a value **WITHOUT UNITS**, e.g. number of cycles etc. the set command is not used so the value is copied using the **equals sign**, e.g.

```
Lastcycle = Cycle
```

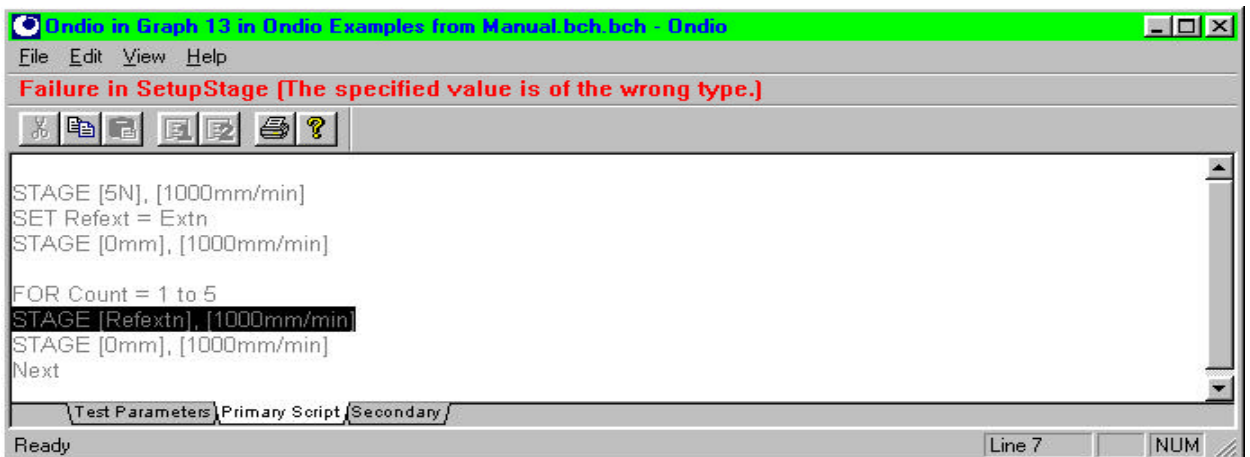
Note that ANY values measured in the Primary script are **ONLY** available in the Primary script. If they are required in the secondary script, they must be stored as a **GLOBAL** variable as shown later in this section.

Using OPTION EXPLICIT and DIM Commands

It is good programming practice to declare variables using the VB command DIM before they are used and to use another VB command **OPTION EXPLICIT** to inform VB not to use any variable that has NOT been declared. This practice ensures that any variables that are spelt incorrectly will be shown with a "User Friendly" error when the test is run. The script below has a spelling mistake on the fifth line where the variable should have been REFEXT. The third STAGE command should move the crosshead to the position measured at the end of the first stage, but the stage limit is undefined because the variable REFEXTN is empty.

```
STAGE [50N], [100mm/min]
SET Refext = Extn.
STAGE [0mm], [100mm/min]
FOR Count = 1 to 5
    STAGE [Refextn], [100mm/min]
    STAGE [0mm], [100mm/min]
NEXT
```

When this test is run, the first two stage will be performed normally but the script will stop on the third stage and the Primary Script screen will be displayed with the error highlight as shown below:-



The error message "**Failure in SetupStage (The specified value is of the wrong type)**" is not easy to understand so to give an easier to understand message, use the OPTION EXPLICIT and DIM commands as shown below:-

```
OPTION EXPLICIT
DIM Refextn
```

The **OPTION EXPLICIT** command **MUST** be the **FIRST LINE** of the script (ignoring any comment lines), or an error will be given when the test is run. The script will now **ONLY** accept Ondio variables e.g. Load, Extn, etc. or any user variables defined using the Dim command. If any variable (Ondio or user) is spelt incorrectly, the script will identify them and stop the test with an easy to understand help message displayed above the toolbar as shown in the example below:-

Note that if a calculation formula is used to define the limit, then any value with units **MUST** be enclosed by square brackets [] and the complete limit statement must also be enclosed by square brackets. Typical units for the stage limit are N, kN, kgf, lbf, mm or in.

The speed may be a constant speed or Load rate control and typical units are mm/min, in/min, N/min, kgf/min and lbf/min. The hold time selects either a creep test (load limit) or a relaxation test (extension limit) and typical units are s, min, hours and days. A complete list of valid units for any limit, speed or results is shown in section 3.20.

3.11 Programming Notes for RESULT

This Ondio command stores a value into the batch table, automatically creating the column if it does not already exist. Example commands are:-

- 1 RESULT "Maximum Load", LoadAtMax**
Stores the Maximum Load value

- 2 RESULT "Marker 1 Load", MarkerLoad(1)**
Stores the load at marker number 1. Note that **at least 1** marker **MUST** be selected on the Test Parameters screen.

- 3 RESULT "Number of Cycles", count**
Stores the number of cycles from a **FOR - NEXT** loop called **COUNT**.
To display the batch result as an **integer**, right click on the "Number of Cycles" column, select Format, Click on Decimal Places then set the Precision to **0**.

- 4 RESULT "Relaxation", [MarkerLoad(1) - MarkerLoad(2)]**
Stores the difference in load between markers number 1 and 2. Note that **at least 2** markers **MUST** be selected on the Test Parameters screen.
Note that the markers contain values **WITH UNITS**, e.g. 100N so the calculation with units **MUST** be fully enclosed by square brackets []

- 5 RESULT "% Elongation", [100 * [Extn] / [Height]]**
Note that if a calculation formula is used, then any value with units **MUST** be enclosed by square brackets [] and the complete result statement must also be enclosed by square brackets.

- 6 RESULT "Maximum Stress", [[LoadAtMax] / [Area]]**
Note that if a calculation formula is used, then any value with units **MUST** be enclosed by square brackets [] and the complete result statement must also be enclosed by square brackets.

- 7 RESULT "Cycle " & count & " Load", Load**
Creates a column title that automatically increments with the value of the **FOR - NEXT** loop called **COUNT** then stores the value of load measured at the end of the last stage. This creates columns called "Cycle 1 Load", "Cycle 2 Load", "Cycle 3 Load" etc. The **&** symbol is used to join (concatenate) text and values **WITHOUT UNITS** into a text string.

- 8 RESULT "Sample Passed", "Yes"**
Creates a "Pass/Fail" indication by storing the text "Yes" into the column. This could be used with an IF - THEN - ELSE statement to check certain measured values then store either "Yes" or "No" as the result.
- 9 RESULT "Sample Passed", TRUE**
Stores a Boolean Value into the column which can be set to "Yes/No" or "Tick/Cross" etc in the batch table. Note that a new column will be created if example 8 is also used in the same batch because the data is different even though the title is the same.
- 10 RESULT "Sample Broke", BROKEN**
Stores a Boolean Value into the column which can be set to "Yes/No" or "Tick/Cross" etc in the batch table. If the sample breaks, the variable BROKEN will be set to TRUE but will stay at FALSE if it did not break. This can also be used to indicate if the Advance or End buttons were pressed using the Boolean variables Advancing or Ending respectively.

3.12 Programming Notes for STATIC and STAGESTATIC

These Ondio commands are used to obtain results from ANY part of the test, i.e. not only at the end of a stage. These commands can be used to obtain data at say 100%, 200% and 300% elongation using only one stage which drives the machine, **WITHOUT PAUSES**, until the sample breaks. These commands can also be used as stage limits. Example commands are:-

- 1 RESULT "Load at 50mm", StaticLoad([50mm])**
Stores the **FIRST** Load value measured at 50mm, i.e. during the first cycle if a cycle test is performed. **Note the use of the round and square brackets.**
- 2 RESULT "Load at 50mm", StageStaticLoad([50mm])**
Stores the Load value measured at 50mm during the LAST stage
- 3 STAGE [StageStaticLoad([50mm]) * 2], [100mm/min]**
Defines the stage limit as being twice the load measured at 50mm during the previous stage (stage speed set to 100mm/min).

3.13 Programming Notes for PROGRESS

This Ondio command displays a message in the status bar during the test. Example commands are:-

- 1 PROGRESS "Cycle " & count**
Displays the cycle number from a **FOR - NEXT** loop called **COUNT**, so the message will change from "Cycle 1", "Cycle 2", "Cycle 3" etc during the test. Note the space between the end of the word cycle and the second " which is required to separate the cycle number from the text, i.e. **Cycle 1**.

- 2 PROGRESS "Load " & LoadAtMax.Name**
 Displays the word "Load" followed by a text string indicating the maximum load value up to this point in the test displayed as text including the "standard" unit of N, e.g. "Load 106.876N"
- 3 PROGRESS "Extension" & Extn.Name**
 Displays the word "Extension" followed by a text string indicating the extension value measured at the end of the last stage displayed as text including the "standard" unit of mm, e.g. "Extension 266.87mm"
- 4 PROGRESS "Cycle " & count & ", Last Max Load " & LoadAtMax.Name**
 Displays the cycle number from a **FOR - NEXT** loop called **COUNT**, together with the text "max load" followed by a text string indicating the maximum load value up to this point in the test. Note the comma and space before the word **Last** and the space after the words **Cycle** and **Load** which are required to give the sentence **Cycle 5, Last Max Load 106.876N**

Note that any variable containing **VALUES WITH UNITS** can be displayed using **.NAME**.

3.14 Programming Notes for MODULUS and OFFSET YIELD

If a Secondary script result requests either the Modulus or an Offset Yield point, e.g. **Result "Stiffness", modulus or Result "Offset Load", LoadAtOffsetYield**, then the steepest slope will be automatically marked on the graph together with the tangential slope line.

The modulus is reported in **Load/Extension** Units, i.e. N/mm, lbf/in etc so is **NOT** the Modulus of Elasticity which is reported in **Stress/Strain** Units, i.e. N/mm², lbf/in². If the Modulus of Elasticity is required, the modulus result has to be converted to MOE by multiplying by Height/Area., e.g. **Result "MOE", [modulus*height/area]**. Note that the height etc contain values **WITH UNITS**, e.g. 50mm so the calculation with units **MUST** be fully enclosed by square brackets [].

The Offset Yield (Proof Stress) results require the Offset Extension to be defined as shown in the example below which calculates the values for 1mm offset extension.

```
SETOFFSETEXTN [1mm]
RESULT "Offset Load", LoadAtOffsetYield
RESULT "Offset Extension", ExtnAtOffsetYield
```

To calculate the 0.2% proof stress, replace the first line with:-
 SETOFFSETEXTN [HEIGHT * 0.002]

To calculate the 5% proof stress, replace the first line with:-
 SETOFFSETEXTN [HEIGHT * 0.05]

3.15 Programming Notes for BREAK Results

The 3 secondary script break results of Loadatbreak, Extnatbreak and Workatbreak will **ONLY** provide valid results **IF THE SAMPLE BROKE** and will give a program error if the sample did not break. If any of these results are required from a test where there is a possibility that the sample may not break, then an **IF BROKEN** statement should be used to ignore the result statement if the sample did not break. If only one break result is required, use a single line as in the example below:-

If Broken then Result "Load at Break", Loadatbreak

If more than one break result is required, use an IF - THEN - ELSE statement as shown below:-

```
If broken then
    Result "Load at Break", Loadatbreak
    Result "Extension at Break", Extnatbreak
End if
```

3.16 Programming Notes for BREAKON and BREAKOFF

The break detector will normally stop a test because when the break detector is triggered, the variable "BROKEN" will be set to TRUE and ALL subsequent STAGE and RUNSTAGE commands in the script will be ignored. Therefore, the test will end because the machine has no further actions to perform, even though the other lines of the script will still be actioned. The break detector can also be used to control the test when it triggers. One type of test that could use this feature is a crush test on a plastic bottle with a cap. The cap is pushed onto the bottle and when it snaps onto the bottle, the bottle has to be compressed by a further 4mm and the crush force measured.

The expected movement to snap the cap on is 3mm so the first stage limit of 10mm is chosen so that stage 2 will be performed after 10mm if the break detector does not trigger for any reason. The break detector **MUST** be selected on the Test Parameters screen and a suitable setting could be manual, 90%.

```
STAGE [10mm], [100mm/min]
BREAKOFF
STAGE [Extn + [4mm] ], [100mm/min]
RESULT "Crush Force", Load
```

The test will start the crosshead moving in the down direction and as soon as the break detector triggers, stage 1 will end and the extension at the "break point" will be automatically stored in the Ondio variable Extn. The break detector sets the variable "BROKEN" to true and causes ALL subsequent STAGE and RUNSTAGE actions to be ignored. The second stage now has to be performed, so the break detector has to be RESET and this is achieved by the BREAKOFF command. The second stage will now be performed and it's limit is the current extension plus 4mm so the crosshead will move down by a further 4mm then the load will be measured and stored in the batch table. Another example is shown in the GLOBAL section later.

3.17 Programming Notes for ADVANCEON and ADVANCEOFF

If a 1000 cycle test has been defined using a FOR - NEXT loop, the operator cannot end the test until all 1000 cycles have been completed. It is sometimes useful to be able to end a test, or part of a test, early but still retain the data stored up to that point. This can be achieved by activating the ADVANCE button which is displayed as a double arrow on the graph toolbar. This button is normally inactive (greyed out) but will become active when the script contains a line with the command ADVANCEON.

If the user clicks on the advance button when it is active, the current stage will end, the advance button will be greyed out again, the variable ADVANCING will be set to TRUE and ALL subsequent STAGE and RUNSTAGE actions will be ignored. This action would therefore end the test in a similar way to a normally ended stage so the values of Load, Extn, TestTime and Work will contain the values when the advance button was clicked as shown in the example below:-

```
ADVANCEON
FOR Cycle = 1 to 1000
    STAGE [25mm], [100mm/min]
    If ADVANCING then exit for
    STAGE [0mm], [100mm/min]
NEXT
RESULT "Stop Load", Load
```

If the test is to continue with another stage, the variable ADVANCING has to be RESET and this can be achieved by using an ADVANCEOFF command as shown in the following example which uses the automatic break detector.

```
BREAKOFF
ADVANCEON
FOR Cycle = 1 to 1000
    STAGE [25mm], [100mm/min]
    If ADVANCING then exit for
    STAGE [0mm], [100mm/min]
NEXT
ADVANCEOFF
BREAKON
STAGE [5kN], [500mm/min]
If broken then RESULT "Break Load", LoadAtBreak
```

The break detector is disabled by the command BREAKOFF and the advance button is enabled by the ADVANCEON command. The sample is cycled for 1000 times then the break detector is switched on by the BREAKON command and the sample is pulled until it breaks.

However, if the user clicks on the Advance button during the cycling, the variable ADVANCING will be set to TRUE and the FOR - NEXT loop will be exited. The test would normally stop because the following STAGE command would be ignored but the ADVANCEOFF command resets the variable ADVANCING to allow the following stage to be actioned.

The variable ADVANCING can also be RESET by using another ADVANCEON command when the advance button is required again during the test as shown in the following example which uses the automatic break detector.

```
BREAKOFF  
ADVANCEON
```

```
FOR Cycle = 1 to 1000  
    STAGE [25mm], [100mm/min]  
    If ADVANCING then exit for  
    STAGE [0mm], [100mm/min]  
NEXT
```

```
ADVANCEON  
STAGE [25mm], [100mm/min]  
FIRSTLOAD = LOAD
```

```
STAGEHOLD [25mm], [100mm/min], [1hours]  
SECONDLOAD = LOAD
```

```
ADVANCEOFF  
BREAKON
```

```
STAGE [5kN], [500mm/min]  
If broken then RESULT "Break Load", LoadAtBreak
```

The break detector is switched off by the command BREAKOFF and the advance button is enabled by the ADVANCEON command. The sample is cycled for 1000 times then a long duration relaxation test is performed on it. At the end of the relaxation period, the break detector is switched on by the BREAKON command and the sample is pulled until it breaks.

However, if the user clicks on the Advance button during the cycling, the variable ADVANCING will be set to TRUE and the FOR - NEXT loop will be exited. All further STAGE commands will now be ignored but the test has to continue with the constant loading section so the STAGE commands are re-enabled using the second ADVANCING command. This second ADVANCEON command not only resets the variable but re-enables the advance button so that the user can click on it again during the relaxation period. The final ADVANCEOFF command allows the final STAGE command to be actioned to break the sample.

However, this method might give a problem if the user double clicks on the advance button during the cycling because the test would exit not only the cycle stages but also the creep stages. This can be overcome by activating the advance button to exit the cycling stages then activating the end button to exit the creep stages.

The same features can be applied to the END button by using the commands ENDON and ENDOFF and the status of the End button is given by the variable ENDING.

3.18 Programming Notes for GLOBAL

Any values measured in the Primary script are **ONLY** available in the Primary script. If they are required in the secondary script, they must be stored as a **GLOBAL** variable in the primary script. The Secondary script is only for results that may require alteration post test, usually a marker point. If the load at the end of one stage is to be compared with the load at a marker point, then the stage load has to be stored in a Global variable as shown below:-

Primary Script

GLOBAL "First", Load

Secondary Script

RESULT "Relaxation", [GLOBAL("First") - MarkerLoad(1)]

Note the square brackets because the calculation involves values **WITH UNITS**.

The example below is a continuation of the BreakOn and BreakOff features

The crush test on a plastic bottle with a cap shown earlier has to report either the crush force at 4mm or the maximum force during the test if the cap did not snap on correctly. This uses the Ondio variable "BROKEN" together with the crush force which has been stored as a GLOBAL variable. The secondary script will only produce one result per test and this will either be the Crush Force if the break detector triggered during the initial part of the test or the maximum force if it did not trigger.

Primary Script

STAGE [10mm], [100mm/min]

BREAKOFF

STAGE [Extn + [4mm]], [100mm/min]

GLOBAL "CF", Load

Secondary Script

```
If BROKEN then
    RESULT "Crush Force", GLOBAL("CF")
else
    RESULT "Maximum Force", LoadAtMax
End If
```

3.19 Programming Notes for User Variables

A User Variable (i.e. not a predefined Ondio Variable) can be one of two types:-

- 1 Containing Numbers Only, i.e. it has **NO UNITS**, e.g. **50**.
- 2 Containing Values (numbers) **WITH UNITS**, e.g. **50N**.

Whenever a variable is created, it will default to numbers only (**NO UNITS**), e.g. **DIM Max**.

If no units are required, then the required value can be stored by a formula e.g. **Max = 10**.

If units **ARE** required, then the required value **WITH UNIT** can be stored using a **SET** command, e.g. **SET Max = [50N]**.

If any **CALCULATIONS** or **COMPARISONS** are to be made concerning Values with Units, then the COMPLETE FORMULA MUST be enclosed in square brackets as shown in the examples below:-

- 1 **[Load + [20N]]**.
Increase the measured load by 20N. Note the square brackets around the 20N.
- 2 **[Load / Area]**
Convert the measured load into a stress
- 3 **[Reflow - Load]**
Subtract the current load from a previously measured reference load
- 4 **If [Load < [50N]] then exit for**
If the current load is less than 50N then exit the For - Next loop.
Do NOT use **[Load = [50N]]** because this will **NEVER** be true and the test will **NOT** end. This principle applies to any measured value.

If a calculation is to be performed on a Value with a Unit, care **MUST** be taken to ensure that the variables and Units are compatible as shown in the examples below:-

- 1 It is **NOT** possible to add a load value to a time value.
- 2 It is **NOT** possible to add Numbers to or subtract Numbers from a variable containing Values with Units, e.g. **Load + 20**. If the variable is to contain a load value which is 20N more than the value held in the Ondio variable "Load", the formula has to be written as **[Load + [20N]]**. Note the square brackets around the value 20N and the square brackets around the complete formula.
- 3 It is **NOT** possible to add a variable containing Values with Units to a variable containing Numbers Only. All variables contain Numbers Only **UNLESS** the **SET** command has been used to convert it to a variable containing Values with Units.

- 4 If a variable containing Values with Units is to be added to a previously unused variable, the second variable **MUST** first be converted from Numbers Only to Values with Units using the **SET** command. This command not only specifies the Units but also specifies the initial value. Generally, an initial value of ZERO is required so to specify a Load variable, store **[0N]** and to specify an Extension variable, store **[0mm]** etc.

The script below calculates the average load over 10 cycles:-

```
DIM Total
```

```
SET Total = [0N]
```

```
For count = 1 to 10
```

```
    STAGE [10mm], [100mm/min]
```

```
    SET Total = Total + Load
```

```
    STAGE [0mm], [100mm/min]
```

```
NEXT
```

```
Result "Average Load", [Total / 10]
```

The variable "**Total**" is set to zero then sequentially adds the measured load values to itself. The average load is calculated by dividing the total load stored by 10 (the number of cycles). Note that a variable containing Values with Units CAN be multiplied or divided by Numbers Only.

Some other examples of using the SET command are shown below:-

- 1 **SET FirstLoad = Load**
Stores the load at the end of the stage in the variable called FirstLoad
- 2 **SET FirstLoad = [0N]**
Specifies that the variable FirstLoad will contain a Load value
Also initialises the variable to [0N]
- 3 **SET FirstLoad = [100N]**
Specifies that the variable FirstLoad will contain a Load value
Also initialises the variable to [100N]
- 4 **SET FirstExt = [0mm]**
Specifies that the variable FirstExt will contain an Extension value
Also initialises the variable to [0mm]

3.20 Tables of Valid Units

Units for Stage Limits or Results				
FORCE	STRESS	EXTENSION	TIME	WORK
N	N/mm ²	µm	ms	Nmm
kN	N/m ²	microns	s	Nm
gf	Pa	mm	min	J
kgf	kPa	cm	hours	kJ
lbf	MPa	m	days	kgf.mm
	lbf/in ²	thous		
		in		

Units for Stage Speeds		
FIXED SPEED (CRT)	FORCE RATE	STRESS RATE
mm/s	N/s	N/mm ² s
mm/min	N/min	N/m ² s
mm/hr	N/hr	MPa/s
cm/min	kN/s	
m/s	kN/min	
m/min	kN/hr	
m/hr	kgf/s	
in/s	kgf/min	
in/min	kgf/hr	
in/hr	lbf/s	
	lbf/min	
	lbf/hr	

Units for Area and other Results				
AREA	VOLUME	MASS	DENSITY	STIFFNESS
mm ²	mm ³	g	g/mm ³	N/mm
cm ²	cm ³	kg	g/cm ³	N/m
m ²	in ³	lb	g/m ³	kN/n
in ²		t		kgf/mm
				lbf/in

To enter an area, either use superscript 2, e.g. [5mm²], or use e.g. [5mm] * [1mm]

To enter a volume, either use superscript 3, e.g. [5mm³], or use e.g. [5mm] * [1mm] * [1mm]

To type the superscript 2, e.g. mm², hold the ALT key then type 253.

To type the superscript 3, e.g. mm³, hold the ALT key then type 252.

To type the µ symbol for micron, e.g. µm, hold the ALT key then type 230.

4. EXAMPLE SCRIPTS

This section contains some examples to demonstrate both the power and flexibility of Ondio and a few graphs have been included only where necessary. The user should select appropriate values for test speeds, limit conditions and sample dimensions etc. to suit the required test. Many comments have been added to the script to indicate the action of the script commands.

Ondio does not use stored setup files but the examples can be cut and pasted into the Ondio Script from an electronic version of this manual.

Assistance with setting up more specialised tests can be obtained by contacting the Technical Support department of Lloyd Instruments Ltd either by fax or email using the address

techsupport@lloyd-instruments.co.uk

The tests in this section are listed below:-

4.1	Pull to Limit	3
4.2	Pull to Break	4
4.3	Pull to Break - Automatic Tangential Modulus	5
4.4	Pull to Break -Tangential Modulus using 2 SetupStages	6
4.5	Pull to Break -Tangential Modulus using Post Test Markers	8
4.6	Pull to Break - Tangential Modulus using StaticExtn Points	10
4.7	Single Cycle to Measure Initial Load	12
4.8	Single Cycle to Measure Initial Permanent Deformation	13
4.9	Cycling to Measure Cycle Loads	14
4.10	Cycling to Measure Average Cycle Loads	16
4.11	Cycling to Measure Cycle Loads using Post Test Markers	18
4.12	Cycling to Measure Cycle Loads - Using the End Button	20
4.13	Cycling to Measure First and Last Cycle Loads	22
4.14	Cycling to Measure First and Last Cycle Loads - Easy to Change No of Cycles	24
4.15	Cycling to Measure First and Last Cycle Loads - Using the End Button	26
4.16	Measuring Energy on 1 Cycle	28
4.17	Measuring Energy on Cycles 1 and 5	30
4.18	Notes for Cycling to Measure Cycle Loads with No Cycle Delay	32
4.19	Cycling to Measure Cycle Loads with No Cycle Delay - Method 1	33
4.20	Cycling to Measure Cycle Loads with No Cycle Delay - Method 2	35

4.21	Cycling to Sample Break	37
4.22	Cycling to Calculated Upper Limit	39
4.23	Cycling until the Sample Stabilises	42
4.24	Cycling until the Sample Weakens	44
4.25	Creep Test - Defined Load	46
4.26	Creep Test - Defined Extension	48
4.27	Relaxation Test - Defined Load	50
4.28	Relaxation Test - Defined Extension	52
4.29	Combined Cycling and Pull to Break	54
4.30	Combined Cycling and Creep Test	56
4.31	Combined Cycling and Relaxation Test	58
4.32	Foam Test 1	60
4.33	Foam Test 2	63
4.34	Foam Test 3	66
4.35	Bottle Top Test 1	69
4.36	Bottle Top Test 2	71
4.37	Pull to Break with Real Time Event Marker	73
4.38	Cycling with Real Time Event Marker	76
4.39	Pausing the Test for Special Purposes	79

4.1 Pull to Limit

A sample is stretched by 50mm and the maximum load and the load at 50mm measured.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not expected
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```

'Example 4.1 - Pull to Limit

Stage [50mm], [100mm/min]
'Alter limit and speed as required

Result "Maximum Load", LoadAtMax

Result "Load at 50mm", Load
'Alter text to suit the limit
    
```

Secondary Script

```

Not used
    
```

Notes:-

The stage limit is set to the required stretch (50mm) and at the end of this stage, the value of load is reported using the variable Load. The maximum load during the test is reported using the variable LoadAtMax.

4.2 Pull to Break

A sample is loaded until it breaks and the maximum load and the load at break measured.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Load drops quickly
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```
'Example 4.2 - Pull to Break

'Ensure break detector is selected on the Test Parameters screen

Stage [50kN], [100mm/min]
'Use a load limit of the loadcell value

Result "Maximum Load", LoadAtMax

'Break result is obtained in the secondary script
```

Secondary Script

```
'Example 4.2 - Pull to Break

If broken then Result "Break Load", LoadAtBreak
```

Notes:-

The stage limit is set to the loadcell value (say 50kN) but this stage will NOT be completed because the sample should break at a lower value. The break detector is required to end the stage when the sample breaks and this is selected on the Test Parameter screen. When the break detector is triggered by the sample breaking, the test will end but the remaining lines of script will be actioned so the maximum load during the test will be reported by the primary script variable LoadAtMax. When the last line of the primary script has been actioned, the secondary script will be actioned and if the sample broke, the Break Load will be reported by the secondary script variable LoadAtBreak.

4.3 Pull to Break with Automatic Tangential Modulus

A sample is loaded until it breaks and the maximum load, load at break and the greatest slope measured.

Test Parameters

Direction	Tension
Preload	None
Height	Exactly 50mm
Area	About 6.0 mm x 3.0 mm
Break	Load drops quickly
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```

'Example 4.3 - Pull to Break with Automatic Tangential Modulus

'Ensure Height, Area and Break Detector are selected on the Test Parameters screen

Stage [50kN], [100mm/min]
'Use a load limit of the loadcell value

Result "Maximum Load", LoadAtMax

'Break and Modulus results are obtained in the secondary script
    
```

Secondary Script

```

'Example 4.3 - Pull to Break with Automatic Tangential Modulus

If broken then Result "Break Load", LoadAtBreak
Result "Modulus of Elasticity", [Modulus * Height / Area]
    
```

Notes:-

The stage limit is set to the loadcell value (say 50kN) but this stage will NOT be completed because the sample should break at a lower value. The break detector is required to end the stage when the sample breaks and this is selected on the Test Parameter screen. The maximum load during the test will be reported by the primary script variable LoadAtMax. When the last line of the primary script has been actioned, the secondary script will be actioned and the Break Load will be reported by the secondary script variable LoadAtBreak. The secondary script variable Modulus will automatically draw the modulus line on the graph. The Modulus of Elasticity is reported by multiplying the modulus result by Height/Area so the height and area MUST be selected on the Test Parameter screen.

4.4 Pull to Break - Tangential Modulus using 2 SetupStages

A sample is loaded until it breaks and the maximum load, load at break and the secant modulus (slope from the origin to a predefined point) measured.

Test Parameters

Direction	Tension
Preload	None
Height	Exactly 50mm
Area	About 6.0 mm x 3.0 mm
Break	Load drops quickly
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```
'Example 4.4 - Pull to Break - Tangential Modulus using 2 SetupStages

'Ensure Height, Area and Break Detector are selected on the Test Parameters screen

SetupStage [1kN], [100mm/min]
'Use a load limit of the required upper secant point - can also use an extension limit

SetupStage [50kN], [100mm/min]
'Use a load limit of the loadcell value

Runstage
'move to upper secant value
Result "Modulus of Elasticity", [ (Load / Extn) * Height / Area]

Runstage
'Continue to the break point
Result "Maximum Load", LoadAtMax

'Break result is obtained in the secondary script
```

Secondary Script

```
'Example 4.4 - Pull to Break - Tangential Modulus using 2 SetupStages

If broken then Result "Break Load", LoadAtBreak
```

Notes for Example 4.4

The test is predefined using 2 SetupStage commands to prevent the slight delay which could occur when the load and extension values are measured at the end of the first stage. The first stage limit is set to the upper secant point and the second stage limit is set to the loadcell value (say 50kN) but this stage will NOT be completed because the sample should break at a lower value. The break detector is required to end the stage when the sample breaks and this is selected on the Test Parameter screen.

The test loads the sample to the first stage limit when the Runstage command is actioned. The load and extension are measured at this point to calculate the sample stiffness. The secant modulus is calculated by multiplying the sample stiffness by Height/Area so the height and area MUST be selected on the Test Parameter screen. Note that NO modulus line will be drawn on the graph.

The maximum load during the test will be reported by the primary script variable LoadAtMax. When the last line of the primary script has been actioned, the secondary script will be actioned and the Break Load will be reported by the secondary script variable LoadAtBreak.

4.5 Pull to Break - Tangential Modulus using Post Test Markers

A sample is loaded until it breaks and the maximum load, load at break and the modulus measured between two post test marker points.

Test Parameters

Direction	Tension
Preload	None
Height	Exactly 50mm
Area	About 6.0 mm x 3.0 mm
Break	Load drops quickly
Auto Zero	Off
Auto Return	On
Markers	2

Primary Script

```
'Example 4.5 - Pull to Break - Tangential Modulus using Post Test Markers

'Ensure Height, Area, Break Detector and 2 markers are selected on the Test Parameters screen

Stage [50kN], [100mm/min]
'Use a load limit of the loadcell value

Result "Maximum Load", LoadAtMax

'Break and modulus results are obtained in the secondary script
```

Secondary Script

```
'Example 4.5 - Pull to Break - Tangential Modulus using Post Test Markers

If broken then Result "Break Load", LoadAtBreak
Set Stiffness = [(Markerload(2) - Markerload(1)) / (Markerextn(2) - Markerextn(1))]
Result "Modulus of Elasticity", [Stiffness * Height / Area]
```

Notes for Example 4.5

The stage limit is set to the loadcell value (say 50kN) but this stage will NOT be completed because the sample should break at a lower value. The break detector is required to end the stage when the sample breaks and this is selected on the Test Parameter screen. The maximum load during the test will be reported by the primary script variable LoadAtMax. When the last line of the primary script has been actioned, the secondary script will be actioned and the Break Load will be reported by the secondary script variable LoadAtBreak. When the test has ended, two markers will be automatically placed at the centre and the end of the graph.

These markers have to be positioned at the required upper and lower modulus points by selecting MOVE MARKER, or click on the MOVE MARKER icon (shown by a down arrow with an asterisk). Position the arrow head over the required marker, press the **LEFT** mouse button, move the marker to the required position then release the **LEFT** mouse button. Note that if the marker is pulled around a major occurrence, e.g. a yield point, a message will be displayed on the Status Line to indicate that the marker cannot be moved this far. To move the marker past this point, press and hold the **CTRL** key as the marker is moved.

After the markers have been positioned as required, the post test variables will be recalculated as shown by the rotation of the Lloyd Logo at the left hand side of the screen. The "Stiffness" result will calculate the sample stiffness in N/mm and the Modulus of Elasticity is calculated by multiplying the Stiffness result by Height/Area so the height and area **MUST** be selected on the Test Parameter screen.

Note that NO modulus line will be drawn on the graph.

4.6 Pull to Break - Tangential Modulus using StaticExtn Points

A sample is loaded until it breaks and the maximum load, load at break and the modulus measured using 2 predefined load points using StaticExtn variables.

Test Parameters

Direction	Tension
Preload	None
Height	Exactly 50mm
Area	About 6.0 mm x 3.0 mm
Break	Load drops quickly
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```
'Example 4.6 - Pull to Break - Tangential Modulus using StaticExtn Points

'Ensure Height, Area and Break Detector are selected on the Test Parameters screen

Stage [50kN], [100mm/min]
'Use a load limit of the loadcell value

Result "Maximum Load", LoadAtMax
'Break and modulus results are obtained in the secondary script
```

Secondary Script

```
'Example 4.6 - Pull to Break - Tangential Modulus using StaticExtn Points

Option Explicit
'traps spelling mistakes of variable names
Dim lower
Dim upper
Dim stress
Dim strain
'declare ALL variables to be used when using option explicit
If broken then Result "Break Load", LoadAtBreak

Set upper = [1000N]
Set lower = [200N]
'define upper and lower LOAD points for measuring the modulus

Set stress = [ (upper - lower) / Area]
Set strain = [ (StaticExtn(upper) - StaticExtn(lower)) / Height]
Result "Modulus of Elasticity", [stress / strain]
```

Notes for Example 4.6

The stage limit is set to the loadcell value (say 50kN) but this stage will NOT be completed because the sample should break at a lower value. The break detector is required to end the stage when the sample breaks and this is selected on the Test Parameter screen. The maximum load during the test will be reported by the primary script variable LoadAtMax. When the last line of the primary script has been actioned, the secondary script will be actioned and the Break Load will be reported by the secondary script variable LoadAtBreak.

When the test has ended, the StaticExtn variables will measure the extension at the load values held in the variables "**upper**" and "**lower**" then the modulus will be calculated. If the modulus is required at different load values, **DOUBLE CLICK** on the Ondio setup, select the Secondary Script then edit the values stored in the "**upper**" and "**lower**" variables as required.

When these have been altered, the post test variables will be recalculated as shown by the rotation of the Lloyd Logo at the left hand side of the screen. The "**Stress**" formula will calculate the stress in MPa and the "**strain**" formula will calculate the strain in percentage elongation so the height and area **MUST** be selected on the Test Parameter screen.

Note that NO modulus line will be drawn on the graph.

Some tests require the modulus to be calculated between two percentages of the maximum load value achieved during the test, e.g. 10% and 30% of the maximum load. This can be achieved by changing the secondary script to:-

'Example 4.6 - Pull to Break - Tangential Modulus using StaticExtn Points

Option Explicit

'traps spelling mistakes of variable names

Dim lower

Dim upper

Dim stress

Dim strain

'declare ALL variables to be used when using option explicit

If broken then Result "Break Load", LoadAtBreak

Set upper = [LoadAtMax * 0.3]

Set lower = [LoadAtMax * 0.1]

'define upper and lower % Max Load points for measuring the modulus

Set stress = [(upper - lower) / Area]

Set strain = [(StaticExtn(upper) - StaticExtn(lower)) / Height]

Result "Modulus of Elasticity", [stress / strain]

4.7 Single Cycle to Measure Initial Load

A sample is stretched by 50mm, the load measured then the sample released.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```

'Example 4.7 - Initial load when cycling

'Ensure that the break detector is switched off

Stage [50mm], [100mm/min]
'Alter limit and speed as required

Result "Load at 50mm", Load
'Alter text to suit the limit

Stage [0mm], [100mm/min]
'Alter limit and speed as required
    
```

Secondary Script

```

Not used
    
```

Notes:-

The first stage limit is set to 50mm and the load at 50mm is reported using the variable Load. The Break Detector is switched off because the test may end too soon if the break detector triggers during the unloading.

4.8 Single Cycle to Measure Initial Permanent Deformation

A sample is stretched by 50mm then released and the increase in length measured immediately after the load has been removed.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```

'Example 4.8 - Permanent Deformation

'Ensure that the break detector is switched off

Stage [50mm], [100mm/min]
'Alter limit and speed as required

Stage [0mm], [100mm/min]
'use a limit of 0mm to return the crosshead to the start of test position

Result "Deformation", StageStaticExtn( [0N] )
'measure the extension when the load falls to 0N during the return stage
'note the use of round and square brackets
    
```

Secondary Script

```

Not used
    
```

Notes:-

The second stage limit is set to 0mm so the crosshead will return to the start of test position. The sample will usually be longer than it's original length so the load will fall to 0N before the crosshead returns to 0mm. The extension where the load falls to 0N is reported using the StageStaticExtn variable. Note that the StaticExtn variable should NOT be used because this will report the extension at the start of the test. The Break Detector is switched off because the test may end too soon if the break detector triggers during the unloading.

Note that the same test can be performed using a second stage limit of **0N** instead of 0mm then reporting the extension using the variable **Extn**. However, the crosshead may bounce at the 0N point if Auto Return is switched on.

4.9 Cycling to Measure Cycle Loads

A sample is stretched by 50mm then released for 5 cycles and the load at the stretch measured for every cycle.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```
'Example 4.9 - Measure load on 5 cycles

'Ensure that the break detector is switched off

Option Explicit
'Traps spelling mistakes of variable names

Dim count
'Declare count as a variable to be used

For count = 1 to 5
'   alter the 5 to the required number of cycles

   progress "Running Cycle No " & count
'   display cycle number on the status bar

   Stage [50mm], [100mm/min]
'   Alter limit and speed as required

   Result "Load at Cycle " & count, Load
'   the column text will be "Load at Cycle 1", "Load at Cycle 2" etc up to "Load at Cycle 5"

   Stage [0mm], [100mm/min]
'   Alter limit and speed as required
Next

'The slight delay at turn round can be minimised by using an advanced setup.
```

Secondary Script

Not used

Notes for Example 4.9

The variable count is declared using the OPTION EXPLICIT and DIM commands.

The first stage limit is set to 50mm and the load at 50mm is reported using the variable Load. The batch row title will be "Load at Cycle 1" because the value of count has been set to 1 during the first cycle by the FOR NEXT loop. The crosshead is returned to the start of test position by the second stage then the value of count is automatically increased to 2 by the FOR NEXT loop. The value of count is less than 5 so the test is repeated and the load at 50mm is reported using the variable Load. The batch row title will be "Load at Cycle 2" because the value of count is now 2. The test repeats for 5 cycles then ends.

The Break Detector is switched off because the test may end too soon if the break detector triggers during the unloading. The slight delay at turn round can be minimised by using an advanced setup.

4.10 Cycling to Measure Average Cycle Loads

A sample is stretched by 50mm then released for 5 cycles and the average load at the stretch measured.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```
'Example 4.10 - Measure Average Cycle Loads

'Ensure that the break detector is switched off

Option Explicit
'Traps spelling mistakes of variable names
Dim count
Dim total
Dim Avg
'Declare count, total and Avg as variables to be used

Set total = [0N]
"Specify that the variable total will contain a load value and set it to 0N

For count = 1 to 5
'  alter the 5 to the required number of cycles
  progress "Cycle " & count
'  display cycle number on the status bar
  Stage [50mm], [100mm/min]
'  Alter limit and speed as required

  Set total = [total + Load]
  Set Avg = [total / count]
  progress "Cycle " & count & ", Avg Load " & Avg.name
'  total contains a value with a unit so is reported using the name property

  Stage [0mm], [100mm/min]
'  Alter limit and speed as required
Next

Result "Average Load", Avg
```

Secondary Script

Not used

Notes for Example 4.10

The variables count and total are declared using the OPTION EXPLICIT and DIM commands. The variable "**total**" will contain a running total of load values so **HAS** to be defined as a load variable using the SET command which is also used to set the initial load to 0N.

The first stage limit is set to 50mm and the load at 50mm is stored in the variable called "total" using the formula **set total = [total + load]**. This formula means "the value stored in the variable total is the original value stored in total plus the measured load value. Note that this calculation is using variables that contain values with units so these must be fully enclosed by square brackets. The average value is calculated using the formula **avg = [total / count]** which also requires the square brackets. The average value is displayed on the status bar using the progress command which outputs a string so the average load value has to be converted to text using the name property. Note the spaces and comma between the quotes to give a correctly formatted text string, e.g. **Cycle 3, Avg Load 123.567N**.

The crosshead is returned to the start of test position by the second stage then the value of count is automatically increased to 2 by the FOR NEXT loop. The value of count is less than 5 so the test is repeated and the load at 50mm is added to the variable total.

The Break Detector is switched off because the test may end too soon if the break detector triggers during the unloading. The slight delay at turn round can be minimised by using an advanced setup.

4.11 Cycling to Measure Cycle Loads using Post Test Markers

A sample is stretched by 50mm then released for 5 cycles and the load at any 2 points measured using post test markers which are positioned by the user after the test has finished.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	2

Primary Script

```
'Example 4.11 - Measure Cycle Loads using Post Test Markers
'Ensure 2 markers are selected on the Test Parameters Screen
'Ensure that the break detector is switched off
For count = 1 to 5
    progress "Running Cycle No " & count
    ' display cycle number on the status bar

    Stage [50mm], [100mm/min]
    ' Alter limit and speed as required

    Stage [0mm], [100mm/min]
    ' Alter limit and speed as required
Next
'The results are calculated in the Secondary Script
```

Secondary Script

```
Result "Load at Marker 1", MarkerLoad(1)
Result "Load at Marker 2", MarkerLoad(2)
Result "Load Difference", [MarkerLoad(1) - MarkerLoad(2)]
'Note the square brackets because of a calculation with units
```

Notes for Example 4.11

The test moves between the first stage limit and the second stage limit 5 times. Note that no results are measurements are made by the Primary Script. The three required results are calculated by the Secondary Script so can be altered post test.

At the end of the test, the 2 markers selected on the Test Parameters screen will be placed on the graph at equal time intervals. These markers can be moved to their required positions by clicking on the "Move Marker" Icon on the toolbar, then the three loads will be calculated and reported using the "MarkerLoad" variables . Note that the "Load Difference" calculation is using variables that contain values with units so must be fully enclosed by square brackets.

The Break Detector is switched off because the test may end too soon if the break detector triggers during the unloading.

The slight delay at turn round can be minimised by using an advanced setup.

4.12 Cycling to Measure Cycle Loads - Using the End Button

A sample is stretched by 50mm then released for 500 cycles and the load at the stretch measured for every cycle. The test can be ended early by clicking on the End Button.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```
'Example 4.12 - Measure load on 500 cycles with End button

'Ensure that the break detector is switched off

Option Explicit
Dim count

EndOn
'Enable the End button

For count = 1 to 500
'   alter the 500 to the required maximum number of cycles

progress "Running Cycle No " & count
'display cycle number on the status bar

    Stage [50mm], [100mm/min]
'   Alter limit and speed as required

    Result "Load at Cycle " & count, Load
'   the column text will be "Load at Cycle 1", "Load at Cycle 2" etc up to "Load at Cycle 5"

    Stage [0mm], [100mm/min]
'   Alter limit and speed as required

    If ending then Exit For
'   prevents unwanted batch columns if the End button is clicked
Next

Result "No of Cycles", Count - 1
```

Secondary Script

Not used

Notes for Example 4.12

The variable count is declared using the OPTION EXPLICIT and DIM commands and the END button is enabled by the ENDON command.

The first stage limit is set to 50mm and the load at 50mm is reported using the variable Load. The batch row title will be "Load at Cycle 1" because the value of count has been set to 1 during the first cycle by the FOR NEXT loop. The crosshead is returned to the start of test position by the second stage then the value of count is automatically increased to 2 by the FOR NEXT loop. The value of count is less than 5 so the test is repeated and the load at 50mm is reported using the variable Load. The batch row title will be "Load at Cycle 2" because the value of count is now 2. The test repeats for 500 cycles then ends.

However, the test will also end if the user clicks on the End button because no more STAGE commands will be actioned, **BUT the script will still continue to run**. If the EXIT FOR command was not used, the script would continue to create batch columns after the test had ended. Therefore, if the test is ended by clicking on the end button, the measured load will be stored in the required column and ALL subsequent columns up to Cycle 500. To prevent these unwanted columns from being filled (and maybe from being created), the EXIT FOR command exits the FOR - NEXT loop and the script continues after the NEXT command.

The number of **COMPLETED** cycles will be stored in the batch by using the formula **count - 1**. To display the batch result as an **integer**, right click on the "No of Cycles" column, select Format, Click on Decimal Places then set the Precision to **0**.

The slight delay at turn round can be minimised by using an advanced setup.

4.13 Cycling to Measure First and Last Cycle Loads

A sample is stretched by 50mm then released for 5 cycles and the load at the stretch measured on the first and last cycle, i.e. cycles 1 and 5.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```
'Example 4.13 - Measure load on first and last defined number of cycles

'Ensure that the break detector is switched off

For count = 1 to 5
'   alter the 5 to the required maximum number of cycles

    progress "Running Cycle No " & count
'   display cycle number on the status bar

    Stage [50mm], [100mm/min]
'   Alter limit and speed as required

    If count = 1 then Result "Load at Cycle 1", Load
    If count = 5 then Result "Load at Cycle 5", Load
'   alter the two 5's to the required maximum number of cycles

    Stage [0mm], [100mm/min]
Next

'The number of cycles can be changed easily by using a different setup

'The slight delay at turn round can be minimised by using an advanced setup
```

Secondary Script

Not used

Notes for Example 4.13

The first stage limit is set to 50mm and the load at 50mm on the first cycle is reported with batch row title of "Load at Cycle 1" using the variable Load. The test repeats for 5 cycles and the load at 50mm on the last cycle is reported with batch row title of "Load at Cycle 5" using the variable Load. The Break Detector is switched off because the test may end too soon if the break detector triggers during the unloading. The slight delay at turn round can be minimised by using a different setup. The number of cycles can be changed easily by using an advanced setup

4.14 Cycling to Measure First and Last Cycle Loads - Easy to Change Number of Cycles

A sample is stretched by 50mm then released for 5 cycles and the load at the stretch measured on the first and last cycle.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```
'Example 4.14 - Measure load on first and last easily changed but defined number of cycles

'Ensure that the break detector is switched off

Option Explicit
'Traps spelling mistakes of variable names

Dim maxcount
Dim count
'Declare ALL variables to be used when using option explicit

maxcount = 5
' alter the 5 to the required maximum number of cycles

For count = 1 to maxcount
  progress "Running Cycle No " & count
  ' display cycle number on the status bar

  Stage [50mm], [100mm/min]
  ' Alter limit and speed as required

  If count = 1 then Result "Load at Cycle 1", Load
  If count = maxcount then Result "Load at Cycle " & maxcount, Load

  Stage [0mm], [100mm/min]
  ' Alter limit and speed as required
Next
```

Secondary Script

Not used

Notes for Example 4.14

The variables maxcount and count are declared using the OPTION EXPLICIT and DIM commands. Note that if OPTION EXPLICIT is used, then ALL user variables MUST be declared using the DIM statement. The maximum number of cycles is defined by the variable maxcount.

The first stage limit is set to 50mm and the load at 50mm on the first cycle is reported with batch row title of "Load at Cycle 1" using the variable Load. The test repeats for the number of cycles set by the variable maxcount and the load at 50mm on the last cycle is reported with batch row title of "Load at Cycle " & maxcount using the variable Load. The Break Detector is switched off because the test may end too soon if the break detector triggers during the unloading. The slight delay at turn round can be minimised by using an advanced setup.

4.15 Cycling to Measure First and Last Cycle Loads - Using the End Button

A sample is stretched by 50mm then released for 500 cycles and the load at the stretch measured on the first and last cycle.. The test can be ended early by clicking on the End Button

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```
'Example 4.15 - Measure load on first and last number of cycles - using the end button

'Ensure that the break detector is switched off

Endon
'Enable the end button

For count = 1 to 500
'   alter the 500 to the required maximum number of cycles
   progress "Running Cycle No " & count
'   display cycle number on the status bar
   Stage [50mm], [100mm/min]
'   Alter limit and speed as required

   If count = 1 then Result "Load at Cycle 1", Load
   If count = 500 then Result "Load at Cycle 500", Load

   If ending then
     If stagenumber/2 = int(stagenumber/2) then
       Result "Load at Cycle " & count, Load
     else
       Result "Load at Cycle " & count - 1, LastLoad
     End If
   Exit For
End If

Set Lastload = Load
Stage [0mm], [100mm/min]
'   Alter limit and speed as required
Next
```

Secondary Script

Not used

Notes for Example 4.15

The END button is enabled by the ENDON command. The first stage limit is set to 50mm and the load at 50mm on the first cycle is reported with batch row title of "Load at Cycle 1" because the value of count is 1. The load value is then copied into a user variable called LASTLOAD then the crosshead returns to 0mm. The test repeats for 500 cycles and the load at 50mm on the last cycle is reported with batch row title of "Load at Cycle 500" because the value of count is 500.

If the end button is pressed, the stagenummer is checked to see if it is an even number or an odd number. If it is an even number, this indicates that the crosshead is returning to the second limit of 0mm. If it is an odd number, this indicates that the crosshead is still moving upwards and the load value that is required is not value from this cycle but the value from the previous cycle which is stored in the variable LASTLOAD.

The check to see if stagenummer is an for odd or even number is given by the formula

$\text{stagenummer}/2 = \text{int}(\text{stagenummer}/2)$.

This will be true for an even number so the load value measured at the end of the upward moving stage will be stored in the batch. If the formula is false, the load value measured on the previous cycle will be stored in the batch. In both cases, the FOR - NEXT loop will be exited if the end button is pressed by the EXIT FOR command.

The Break Detector is switched off because the test may end too soon if the break detector triggers during the unloading. The slight delay at turn round can be minimised by using an advanced setup.

4.16 Measuring Energy on 1 Cycle

A sample is stretched by 50mm then released and the energy measured.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```
'Example 4.16 - Measure Energy on 1 cycle

'Ensure that the break detector is switched off

Option Explicit
'Traps spelling mistakes of variable names

Dim Up
Dim Down
'Declare ALL variables to be used when using option explicit

Stage [50mm], [100mm/min]
'Alter limit and speed as required

Set Up = Work
'Store energy during first stage from start of test - energy to stretch sample

Stage [0mm], [100mm/min]
'Alter limit and speed as required

Set Down = Work
'Store total energy during cycle from start of test - energy absorbed by sample

Result "Energy to stretch Sample", Up
Result "Energy returned by Sample", [Up - Down]
'Note the square brackets because of a calculation with units

Result " Energy absorbed by Sample", Down
```

Secondary Script

Not used

Notes for Example 4.16

The first stage limit is set to 50mm and the energy used to stretch the sample to 50mm is measured and stored in the user variable called "Up" by the SET command which is required because the variable Work contains a value with units. The second stage limit is 0mm and the TOTAL cumulative energy during the cycle is measured and stored in the user variable called "Down". Note that this cumulative energy is both positive (energy required to stretch the sample) and negative (energy given up by the sample) so the TOTAL energy after one cycle will be close to zero if the sample is very elastic and this is the energy absorbed by the sample.

If the energy given up by the sample is required, this can be calculated from the difference in energy between the stretch and the complete cycle and is given by [Up - Down]. Note that this calculation is using variables that contain values with units so must be fully enclosed by square brackets.

The slight delay at turn round can be minimised by using an advanced setup.

4.17 Measuring Energy on Cycles 1 and 5

A sample is stretched by 50mm then released for 5 cycles and the energy absorbed by the sample is measured on the first and the last cycles.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```
'Example 4.17 - Measure Energy on Cycles 1 and 5

'Ensure that the break detector is switched off

Option Explicit
'Traps spelling mistakes of variable names

Dim count
Dim Energy4
'Declare ALL variables to be used when using option explicit

For count = 1 to 5
  progress "Running Cycle No " & count
  ' display cycle number on the status bar

  Stage [50mm], [100mm/min]
  ' Alter limit and speed as required

  Stage [0mm], [100mm/min]
  ' Alter limit and speed as required

  If count = 1 then Result "Energy absorbed on cycle 1", Work
  ' Report Energy from start of test to the end of the first cycle
  If count = 4 then Set Energy4 = Work
  ' Store Energy from start of test to the end of the fourth cycle
Next

Result "Energy absorbed on cycle 5", [Work - Energy4]
'Note the square brackets because of a calculation with units
```

Secondary Script

Not used

Notes for Example 4.17

The variables count and Energy4 are declared using the OPTION EXPLICIT and DIM commands. Note that if OPTION EXPLICIT is used, then ALL user variables MUST be declared using the DIM statement.

The first stage limit is set to 50mm and the second stage limit is 0mm. The TOTAL cumulative energy during the cycle is measured and stored in the batch table from the variable Work. The sample is cycled 3 more times and at the END of fourth cycle, the TOTAL cumulative energy from the start of test is measured and stored in the user variable called "Energy4" by the SET command which is required because the variable Work contains a value with units.

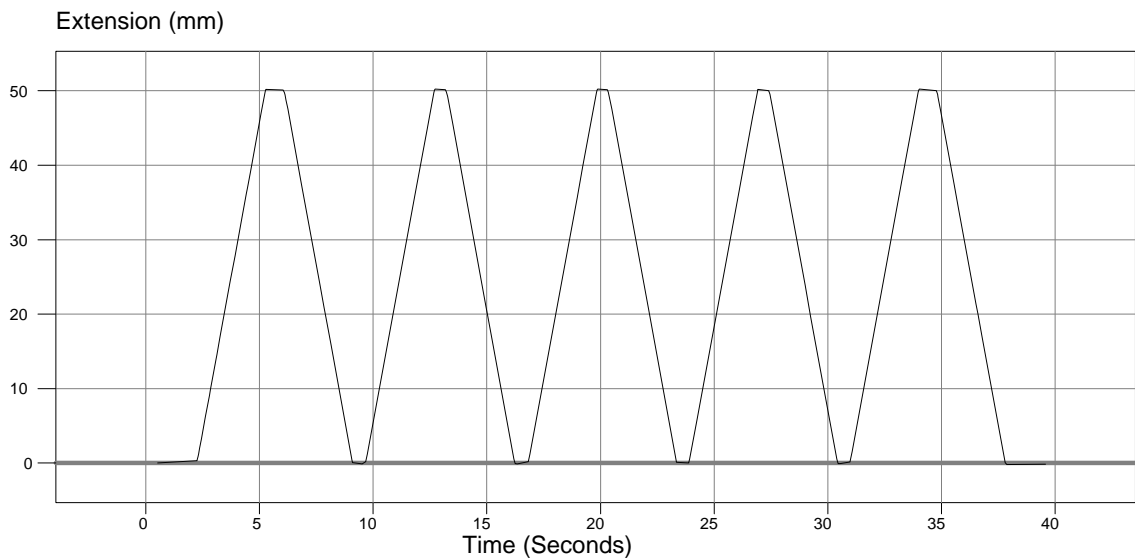
The sample is cycled one more time and at the END of fifth cycle, the TOTAL cumulative energy from the start of test is measured and stored in the Ondio variable Work. The energy expended during the fifth cycle is calculated from the formula $[Work - Energy4]$. Note that this calculation is using variables that contain values with units so must be fully enclosed by square brackets.

The slight delay at turn round can be minimised by using an advanced setup.

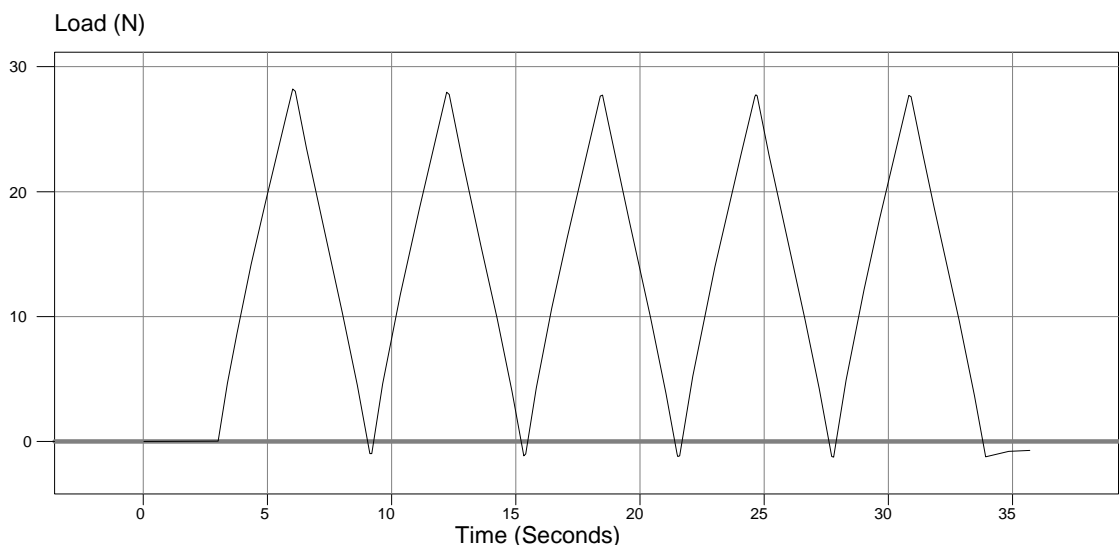
4.18 Notes for Cycling to Measure Cycle Loads with No Cycle Delay

The slight delay at turn round does not usually cause any problems, but if there is a specific reason why the crosshead should change direction without a delay, then the script can be modified to prevent this. There are several ways of writing a script to achieve this and the following scripts are included in this manual to show the principle. There is no right way or wrong way of writing any script. The simpler scripts are easier to write and understand but may have a long start up delay if the number of cycles exceeds approx. 50. The more advanced scripts will work for any number of cycles.

To eliminate any turn round delay during the cycling, the machine always has to have at least two stages loaded:- the current stage to be performed and the stage that has to be performed immediately after the current stage has ended. This is achieved by loading the stages using the SETUPSTAGE command then running the stages later by using the RUNSTAGE command.



A test using STAGES has a slight delay at turn-round



A test using SETUPSTAGES and RUNSTAGES has minimal delay at turn-round

4.19 Cycling to Measure Cycle Loads with No Cycle Delay - Method 1

A sample is stretched by 50mm then released for 5 cycles and the load at the stretch measured for every cycle.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

Primary Script

```
'Example 4.19 - Measure load on 5 cycles - no cycle delay - method 1

'Ensure that the break detector is switched off

Option Explicit
'Traps spelling mistakes of variable names
Dim count
Dim maxcount
'Declare ALL variables to be used when using option explicit

maxcount = 5
'alter the 5 to the required maximum number of cycles

For count = 1 to maxcount
    SetupStage [50mm], [100mm/min]
    SetupStage [0mm], [100mm/min]
    ' Send the cycle information to the machine but do not start the test
Next

For count = 1 to maxcount
    progress "Running Cycle No " & count
    ' display cycle number on the status bar

    Runstage
    ' Run stage 1 to first limit
    Result "Load at Cycle " & count, Load
    ' the column text will be "Load at Cycle 1", "Load at Cycle 2" etc up to "Load at Cycle 5"
    Runstage
    ' Run stage 2 to second limit
Next
```

Secondary Script

Not used

Notes for Example 4.19

In order to create an easily modified setup, the maximum number of cycles is defined using the variable maxcount. This setup sends ALL the stages required to perform the complete test to the machine using the 2 SETUPSTAGE commands together with the first FOR - NEXT loop. The test is performed by running the stages using the 2 RUNSTAGE commands together with the second FOR - NEXT loop.

The first RunStage command will perform the first stage and the second RunStage will perform the second stage with NO delay between stages and the test repeats for 5 cycles then ends. The value of load is measured at the end of the first stage and is reported with the title "Load at Cycle 1" etc.

The Break Detector is switched off because the test may end too soon if the break detector triggers during the unloading.

Note that there will be an initial delay before the test starts and this is due to the time it takes to load the 10 stages to the machine using the SETUPSTAGE commands. The delay will depend upon the speed of the computer but will be approx. 10 seconds if a 50 cycle test is to be performed. **Therefore, this method is only recommended for a small number of cycles.**

4.20 Cycling to Measure Cycle Loads with No Cycle Delay - Method 2

A sample is stretched by 50mm then released for 5 cycles and the load at the stretch measured for every cycle.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

Not used

Notes for Example 4.20

In order to create an easily modified setup, the maximum number of cycles, the cycle Upper Limit and the cycle Lower Limit are all defined using the variables maxcount, UpperLimit and LowerLimit. Note that the limit variables have to be defined using the SET command because these variables contain values with units. Also note that when the variables UpperLimit and LowerLimit are used in the STAGE commands, they are not enclosed in square brackets.

The first cycle (2 stages) are sent to the machine using the 2 SETUPSTAGE commands then the FOR - NEXT loop is entered. The FOR - NEXT loop also sends the same 2 stages to the machine so this now has 4 stages to run. The first RunStage command will perform the first stage and the second RunStage will perform the second stage with NO delay between stages. The machine still has 2 stages left to run but the FOR - NEXT loop repeats and sends another 2 stages to the machine.

The FOR - NEXT loop is repeated until all cycles EXCEPT THE LAST CYCLE is run. The machine still has 2 stages left to run and these are performed by the last 2 RUNSTAGE commands after the FOR - NEXT loop.

The Break Detector is switched off because the test may end too soon if the break detector triggers during the unloading.

Note that there the initial delay before the test starts is not dependant upon the number of cycles because only 4 stages have to be sent to the machine before the test starts.

Therefore, this method is recommended for a large number of cycles.

Primary Script

```
'Example 4.20 - Measure load on 5 cycles - no cycle delay - method 2

Option Explicit
'Traps spelling mistakes of variable names
Dim count
Dim maxcount
Dim UpperLimit
Dim LowerLimit
'Declare ALL variables to be used when using option explicit

maxcount = 5
'alter the 5 to the required maximum number of cycles

Set UpperLimit = [50mm]
'Alter the [50mm] to the required Upper Limit
Set LowerLimit = [0mm]
'Alter the [0mm] to the required Lower Limit

SetupStage UpperLimit, [100mm/min]
SetupStage LowerLimit, [100mm/min]
'Send the first cycle information to the machine but do not start the test

For count = 1 to maxcount - 1
    SetupStage UpperLimit, [100mm/min]
    SetupStage LowerLimit, [100mm/min]
    ' Send next cycle information to the machine but do not run

    progress "Running Cycle No " & count
    ' display cycle number on the status bar

    Runstage
    ' Run stage 1 to first limit
    Result "Load at Cycle " & count, Load
    ' the column text will be "Load at Cycle 1", "Load at Cycle 2" etc up to "Load at Cycle 4"
    Runstage
    ' Run stage 2 to second limit
Next

'Have now run all but last cycle
progress "Running Cycle No " & maxcount
'display cycle number on the status bar

Runstage
Result "Load at Cycle " & maxcount, Load
'the column text will be "Load at Cycle " & maxcount
Runstage
```

4.21 Cycling to Sample Break

A sample is repeatedly stretched by 50mm then released until it breaks and the break load, extension at break and the number of cycles to break measured.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Load drops quickly
Auto Zero	Off
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

```

'Example 4.21 - Cycling to Sample Break

If broken then
  Result "Break Load", Loadatbreak
  Result "Break Extension", Extnatbreak
End if
    
```

Notes for Example 4.21

The required results are the break values so the break detector **MUST** be selected on the Test Parameters screen. However, when performing a cyclic test, the test may end when the load falls during the unloading stage depending upon the type of detector used and the speed that the load drops. To ensure that the break detector does not end the test incorrectly, disable it during each unloading stage using the BREAKOFF command and enable it during each loading stage using the BREAKON command.

The secondary script break results of Loadatbreak and Extnatbreak will **ONLY** provide valid results **IF THE SAMPLE BROKE** and will give a program error if the sample did not break. If the test ends by performing the maximum number of cycles, or if the user clicks on the Abort Test button, the secondary script will give a program error. To prevent the possibility of this error occurring, the **IF BROKEN** statement is used to ignore the results if the sample did not break.

The number of **COMPLETED** cycles will be stored in the batch by using the formula count - 1. To display the batch result as an **integer**, right click on the "Number of Cycles" column, select Format, Click on Decimal Places then set the Precision to **0**.

The slight delay at turn round can be minimised by using an advanced setup.

Primary Script

```
'Example 4.21 - Cycling to Sample Break

'Ensure break detector is selected on the Test Parameters screen

Option Explicit
'Traps spelling mistakes of variable names

Dim count
'Declare the variables to be used when using option explicit

For count = 1 to 10000
'use a high number to allow sample to break - can also be used as a safety stop value

    breakon
    ' activate break detector - has no effect on first cycle

    progress "Running Cycle No " & count
    ' display cycle number on the status bar

    Stage [50mm], [100mm/min]
    ' Alter limit and speed as required

    if broken then exit for

    breakoff
    ' disable break detector when removing the load from sample

    Stage [0mm], [100mm/min]
    ' Alter limit and speed as required
Next

Result "No of Cycles", count - 1
'Number of COMPLETED cycles

'Break results are obtained in the secondary script

"The slight delay at turn round can be minimised by using an advanced setup
```

4.22 Cycling to Calculated Upper Limit

A sample is repeatedly stretched and released until it breaks by subjecting it to a successively increasing crosshead travel, e.g. 10mm, 20mm 30mm etc. The break load, extension at break and the number of cycles to break are measured.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Load drops quickly
Auto Zero	Off
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

```
'Example 4.22 - Cycling to Calculated Upper Limit
If broken then
    Result "Break Load", Loadatbreak
    Result "Break Extension", Extnatbreak
End if
```

Notes for Example 4.22

The required results are the break values so the break detector **MUST** be selected on the Test Parameters screen. However, when performing a cyclic test, the test may end when the load falls during the unloading stage depending upon the type of detector used and the speed that the load drops. To ensure that the break detector does not end the test incorrectly, disable it during each unloading stage using the BREAKOFF command and enable it during each loading stage using the BREAKON command.

The secondary script break results of Loadatbreak and Extnatbreak will **ONLY** provide valid results **IF THE SAMPLE BROKE** and will give a program error if the sample did not break. If the test ends by performing the maximum number of cycles, or if the user clicks on the Abort Test button, the secondary script will give a program error. To prevent the possibility of this error occurring, the **IF BROKEN** statement is used to ignore the results if the sample did not break.

The variable "**travel**" contains a constant extension value so **HAS** to be defined as an extension variable using the SET command which is also used to set the initial travel to 10mm. The upperlimit is a calculated extension which is used as the limit condition for the first stage. The number of **COMPLETED** cycles will be stored in the batch by using the formula **count - 1**. To display the batch result as an **integer**, right click on the "Number of Cycles" column, select Format, Click on Decimal Places then set the Precision to **0**.

Primary Script

```
'Example 4.22 - Cycling to Calculated Upper Limit

'Ensure break detector is selected on the Test Parameters screen

Option Explicit
'Traps spelling mistakes of variable names

Dim count
Dim upperlimit
Dim travel
'Declare ALL variables to be used when using option explicit

Set travel = [10mm]
'Specify that the variable travel will contain an extension value
'Set the initial extension and cycle increment to 10mm

for count = 1 to 50
'Use a high enough number to allow sample to break
'Can also be used as a safety stop value - 50 increases of 10mm gives 500mm maximum travel

    breakon
    ' activate break detector - has no effect on first cycle

    Set upperlimit = [travel * count]
    ' Calculate the limit value

    progress "Moving to " & upperlimit.name
    ' display upper limit value on the status bar - note the name property

    Stage upperlimit, [100mm/min]
    ' Alter speed as required

    if broken then exit for

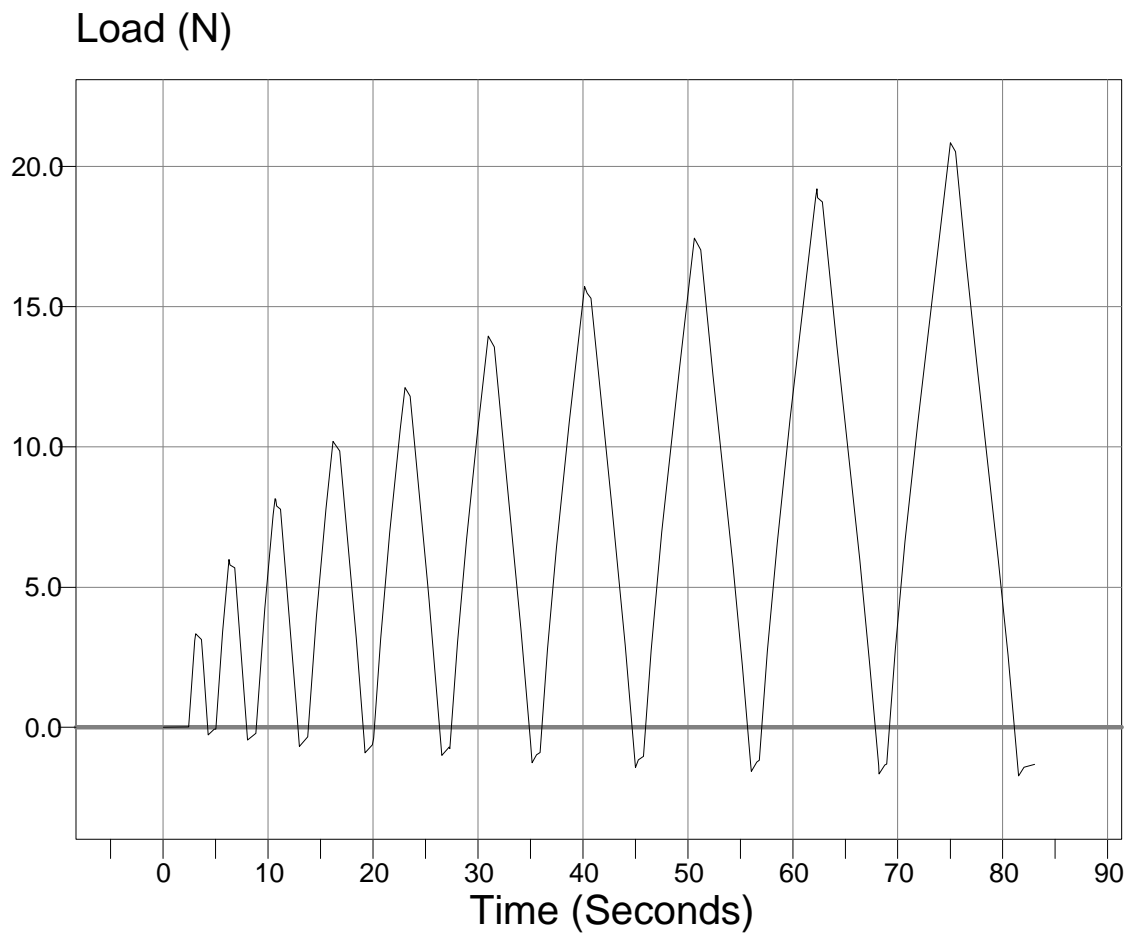
    breakoff
    ' disable break detector when removing the load from sample

    Stage [0mm], [100mm/min]
    ' Alter limit and speed as required
Next

Result "No of Cycles", count - 1

'Break results are obtained in the secondary script

'The slight delay at turn round can be minimised by using an advanced setup.
```

Graph for Example 4.22

4.23 Cycling until the Sample Stabilises

A sample is loaded by 50N, the extension measured, then the load removed. The sample is then repeatedly cycled until the increase in sample length between cycles is less than 0.1mm then the number of cycles required to achieve sample stabilisation is reported.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

Not used

Notes for Example 4.23

The variable "**lastext**" will contain an extension value so **HAS** to be defined as an extension variable using the SET command which is also used to set the initial extension to 0mm.

The sample is loaded to 50N and the extension stored in the variable "**refext**". The difference between this extension and the extension stored in the variable "**lastext**" (0mm at the start of the test) is calculated and stored in the variable "**difference**". This value is also displayed on the status bar to show how the sample is stretching during the test. The sample is then unloaded and the value stored in the variable "**difference**" is checked. If this value is greater than 0.1mm, then the extension value stored in the variable "**refext**" is copied into the variable "**lastext**" and the cycle is repeated.

The sample is loaded again to 50N and the extension stored in the variable "**refext**". The difference between this extension and the extension stored in the variable "**lastext**" (the value which was stored in "**refext**" during the last stretch) is calculated and stored in the variable "**difference**". The sample is then unloaded and the value stored in the variable "**difference**" is checked. The test repeats until the value stored in the variable "difference" falls below 0.1mm then the test will end. Note that the comparison formula [**difference** < **[0.1mm]**] requires square brackets because the variable "**difference**" contains VALUES WITH UNITS.

The number of **COMPLETED** cycles will be stored in the batch by using the formula **count**. To display the batch result as an **integer**, right click on the "Number of Cycles" column, select Format, Click on Decimal Places then set the Precision to **0**.

Primary Script

'Example 4.23 - Cycling until the Sample Stabilises

'Ensure that the break detector is switched off

Option Explicit

'Traps spelling mistakes of variable names

Dim count

Dim refext

Dim difference

Dim lasttext

'Declare ALL variables to be used when using option explicit

Set lasttext = [0mm]

"Specify that the variable lasttext will contain an extension value and set it to 0mm

For count = 1 to 10000

'use a high number to allow sample to break - can also be used as a safety stop value

 Stage [50N], [100mm/min]

' Alter limit and speed as required

 Set refext = Extn

 Set difference = [Extn - lasttext]

 progress "Increase " & difference.name

' display difference value on the status bar - note the name property

 Stage [0mm], [100mm/min]

' Alter limit and speed as required

 If [difference < [0.1mm]] then exit for

' Note the brackets around the comparison formula

 Set lasttext = refext

Next

Result "No of Cycles", count

'The slight delay at turn round can be minimised by using an advanced setup.

4.24 Cycling until the Sample Weakens

A carton is compressed by a load of 50N, the extension measured then the load removed. The sample is then repeatedly cycled until its height at 50N is reduced by more than 5mm then the number of cycles required to achieve sample stabilisation reported.

Test Parameters

Direction	Compression
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

Not used

Notes for Example 4.24

The variable "**lastext**" will contain an extension value so **HAS** to be defined as an extension variable using the SET command which is also used to set the initial extension to 0mm.

The sample is loaded to 50N and the extension stored in the variable "**refext**". The difference between this extension and the extension stored in the variable "**lastext**" (0mm at the start of the test) is calculated and stored in the variable "**difference**". This value is also displayed on the status bar to show how the sample is stretching during the test. The sample is then unloaded and the value stored in the variable "**difference**" is checked. If this value is greater than 0.1mm, then the extension value stored in the variable "**refext**" is copied into the variable "**lastext**" and the cycle is repeated.

The sample is loaded again to 50N and the extension stored in the variable "**refext**". The difference between this extension and the extension stored in the variable "**lastext**" (the value which was stored in "**refext**" during the last stretch) is calculated and stored in the variable "**difference**". The sample is then unloaded and the value stored in the variable "**difference**" is checked. The test repeats until the value stored in the variable "difference" falls below 0.1mm then the test will end. Note that the comparison formula [**difference** < **[0.1mm]**] requires square brackets because the variable "**difference**" contains VALUES WITH UNITS.

The number of **COMPLETED** cycles will be stored in the batch by using the formula **count**. To display the batch result as an **integer**, right click on the "Number of Cycles" column, select Format, Click on Decimal Places then set the Precision to **0**.

Primary Script

'Example 4.24 - Cycling until the Sample Weakens

'Ensure that Compression is selected and that the break detector is switched off

Option Explicit

'Traps spelling mistakes of variable names

Dim count

Dim refext

Dim difference

'Declare ALL variables to be used when using option explicit

For count = 1 to 10000

'use a high number to allow sample to break - can also be used as a safety stop value

 Stage [50N], [100mm/min]

' Alter limit and speed as required

 If count = 1 then Set refext = Extn

' Store the initial compression

 Set difference = [Extn - refext]

 progress "Compression " & difference.name

' display difference value on the status bar - note the name property

 Stage [0mm], [100mm/min]

' Alter limit and speed as required

 If [difference > [5mm]] then exit for

' Note the brackets around the comparison formula

Next

Result "No of Cycles", count

'The slight delay at turn round can be minimised by using an advanced setup.

4.25 Creep Test - Defined Load

A sample is loaded to 50N then this load maintained for a 10 minute time period. The increase in extension (sample creep) over this time period is required.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

Not used

Notes for Example 4.25

The sample is loaded to 50N and the increase in extension (sample creep) measured over a 10 minute time period. The End button is enabled to allow the operator to end the test early and a "Test Stopped" result is provided to indicate whether the test was stopped by the end button.

The required load value is stored in the variable "**limit**" and the required hold time is stored in the variable "**holdtime**". The first stage applies the load then measures the extension and the time as soon as the load increases to the load value. The end button is then enabled and the second stage maintains the load for the specified hold time. The extension is measured at the end of the hold time and the difference in extension given as a result together with the specified hold time.

If the user stops the test by clicking on the End button, the variable "**holdtime**" will store the actual holdtime instead of the specified holdtime. The result "**Test Stopped**" reports a Boolean result which will be TRUE if the user clicked on the end button or FALSE if the test continued to the end of the specified hold time. This result can be set to YES/NO, TRUE/FALSE etc by setting the properties of the batch column.

The progress command is used to indicate that the machine is either moving to apply the required load or is maintaining the load.

Primary Script

```
'Example 4.25 - Creep Test - Defined Load

'Ensure that the break detector is switched off

Option Explicit
'Traps spelling mistakes of variable names

Dim limit
Dim holdtime
Dim refext
Dim reftime
'Declare ALL variables to be used when using option explicit

Set limit = [50N]
'Specify that the variable limit will contain a load value and set it to 50N
Set holdtime = [10min]
'Specify that the variable holdtime will contain a time value and set it to 10min

progress "Moving to " & limit.name
'Display message on status bar - note the name property

Stage limit, [100mm/min]
'Move to load limit
'Alter speed as required

Set refext = Extn
Set reftime = Testtime
'Store extension and time at the beginning of the constant loading

Endon
'Enable the End button

Progress "Maintaining " & limit.name
'Display message on status bar - note the name property

Stagehold limit, [100mm/min], holdtime
'Apply constant load
'Alter speed as required

If ending then Set holdtime = [Testtime - reftime]
'report actual hold time if test stopped by pressing the End button'

Result "Sample Creep", [Extn - refext]
Result "Hold Time", holdtime
Result "Test Stopped", Ending
'Will give result of TRUE if test stopped by pressing the End button or FALSE if not
```

4.26 Creep Test - Defined Extension

A sample is stretched by 100mm, the load measured then this load maintained for a 10 minute time period. The increase in extension (sample creep) over this time period is required.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

Not used

Notes for Example 4.26

The sample is stretched by 100mm then the load at this extension maintained for a 10 minute time period. The End button is enabled to allow the operator to end the test early and a "Test Stopped" result is provided to indicate whether the test was stopped by the end button.

The required extension value is stored in the variable "**limit**" and the required hold time is stored in the variable "**holdtime**". The first stage stretches the sample by the required extension then measures the load and the time at this stretch. The end button is then enabled and the second stage maintains the measured load for the specified hold time. The extension is measured at the end of the hold time and the difference in extension given as a result together with the specified hold time.

If the user stops the test by clicking on the End button, the variable "**holdtime**" will store the actual holdtime instead of the specified holdtime. The result "**Test Stopped**" reports a Boolean result which will be TRUE if the user clicked on the end button or FALSE if the test continued to the end of the specified hold time. This result can be set to YES/NO, TRUE/FALSE etc by setting the properties of the batch column.

The progress command is used to indicate that the machine is either moving to stretch the sample or is applying the required load or is maintaining the load.

Primary Script

```
'Example 4.26 - Creep Test - Defined Extension

'Ensure that the break detector is switched off

Option Explicit
'Traps spelling mistakes of variable names

Dim limit
Dim holdtime
Dim reftime
'Declare ALL variables to be used when using option explicit

Set limit = [100mm]
'Specify that the variable limit will contain an extension value and set it to 100mm

Set holdtime = [10min]
'Specify that the variable holdtime will contain a time value and set it to 10min

progress "Moving to " & limit.name
'Display message on status bar - note the name property

Stage limit, [100mm/min]
'Move to extension limit
'Alter speed as required

Set reftime = Testtime
'Store time at the beginning of the constant loading

Endon
'Enable the End button

Progress "Maintaining " & load.name
'Display load value on status bar - note the name property

Stagehold load, [100mm/min], holdtime
'Apply constant load as measured at the end of the first stage
'Alter speed as required

If ending then Set holdtime = [Testtime - reftime]
'report actual hold time if test stopped by pressing the End button'

Result "Sample Creep", [Extn - limit]
Result "Hold Time", holdtime
Result "Test Stopped", Ending
'Will give result of TRUE if test stopped by pressing the End button or FALSE if not
```

4.27 Relaxation Test - Defined Load

A sample is loaded to 50N, the extension measured then this extension maintained for a 10 minute time period. The decrease in load (sample relaxation) over this time period is required.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

Not used

Notes for Example 4.27

The sample is loaded to 50N, the extension at this load measured then maintained for a 10 minute time period. The End button is enabled to allow the operator to end the test early and a "Test Stopped" result is provided to indicate whether the test was stopped by the end button.

The required load value is stored in the variable "**limit**" and the required hold time is stored in the variable "**holdtime**". The first stage applies the required load then measures the extension and the time at this stretch. The end button is then enabled and the second stage holds the crosshead at the measured extension for the specified hold time. The load is measured at the end of the hold time and the difference in load given as a result together with the specified hold time.

If the user stops the test by clicking on the End button, the variable "**holdtime**" will store the actual holdtime instead of the specified holdtime. The result "**Test Stopped**" reports a Boolean result which will be TRUE if the user clicked on the end button or FALSE if the test continued to the end of the specified hold time. This result can be set to YES/NO, TRUE/FALSE etc by setting the properties of the batch column.

The progress command is used to indicate that the machine is either moving to stretch the sample or is holding the crosshead at the required extension.

Primary Script

'Example 4.27 - Relaxation Test - Defined Load

'Ensure that the break detector is switched off

Option Explicit

'Traps spelling mistakes of variable names

Dim limit

Dim holdtime

Dim reftime

'Declare ALL variables to be used when using option explicit

Set limit = [50N]

'Specify that the variable limit will contain a load value and set it to 50N

Set holdtime = [10min]

'Specify that the variable holdtime will contain a time value and set it to 10min

progress "Moving to " & limit.name

'Display message on status bar - note the name property

Stage limit, [100mm/min]

'Move to load limit

'Alter speed as required

Set reftime = Testtime

'Store time at the beginning of the relaxation

Endon

'Enable the End button

Progress "Maintaining " & Extn.name

'Display extension value on status bar - note the name property

Stagehold Extn, [100mm/min], holdtime

'keep crosshead at extension value measured at the end of the first stage

'Alter speed as required

If ending then Set holdtime = [Testtime - reftime]

'report actual hold time if test stopped by pressing the End button'

Result "Relaxation", [limit - Load]

'note that load falls during the hold time

Result "Hold Time", holdtime

Result "Test Stopped", Ending

'Will give result of TRUE if test stopped by pressing the End button or FALSE if not

4.28 Relaxation Test - Defined Extension

A sample is stretched by 100mm and this extension maintained for a 10 minute time period. The decrease in load (sample relaxation) over this time period is required.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

Not used

Notes for Example 4.28

The sample is stretched by 100mm then the crosshead maintained at this extension for a 10 minute time period. The End button is enabled to allow the operator to end the test early and a "Test Stopped" result is provided to indicate whether the test was stopped by the end button.

The required extension value is stored in the variable "**limit**" and the required hold time is stored in the variable "**holdtime**". The first stage stretches the sample by the required extension then measures the load and the time at this stretch. The end button is then enabled and the second stage maintains the crosshead position for the specified hold time. The load is measured at the end of the hold time and the difference in load given as a result together with the specified hold time.

If the user stops the test by clicking on the End button, the variable "**holdtime**" will store the actual holdtime instead of the specified holdtime. The result "**Test Stopped**" reports a Boolean result which will be TRUE if the user clicked on the end button or FALSE if the test continued to the end of the specified hold time. This result can be set to YES/NO, TRUE/FALSE etc by setting the properties of the batch column.

The progress command is used to indicate that the machine is either moving to stretch the sample or is holding the crosshead at the required extension.

Primary Script

'Example 4.28 - Relaxation Test - Defined Extension

'Ensure that the break detector is switched off

Option Explicit

'Traps spelling mistakes of variable names

Dim limit

Dim holdtime

Dim reflow

Dim reftime

'Declare ALL variables to be used when using option explicit

Set limit = [100mm]

'Specify that the variable limit will contain an extension value and set it to 100mm

Set holdtime = [10min]

'Specify that the variable holdtime will contain a time value and set it to 10min

progress "Moving to " & limit.name

'Display message on status bar - note the name property

Stage limit, [100mm/min]

'Move to extension limit

'Alter speed as required

Set reflow = Load

Set reftime = Testtime

'Store load and time at the beginning of the relaxation

Endon

'Enable the End button

Progress "Maintaining " & limit.name

'Display message on status bar - note the name property

Stagehold load, [100mm/min], holdtime

'keep crosshead at limit extension value

'Alter speed as required

If ending then Set holdtime = [Testtime - reftime]

'report actual hold time if test stopped by pressing the End button'

Result "Relaxation", [reflow - Load]

'note that the load falls during the hold time

Result "Hold Time", holdtime

Result "Test Stopped", Ending

'Will give result of TRUE if test stopped by pressing the End button or FALSE if not

4.29 Combined Cycling and Pull to Break

A sample is stretched by 50mm then released for up to 50 cycles then it is pulled until it breaks. The results required are the Load and Extension at Break. The cyclic part of the test can be ended early by clicking on the Advance Button and this will be indicated by the Number of Cycles Result.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Load drops quickly
Auto Zero	Off
Auto Return	On
Markers	0

The Primary Script is shown on the next page
Secondary Script

'Example 4.29 - Combined Cycling and Pull to Break

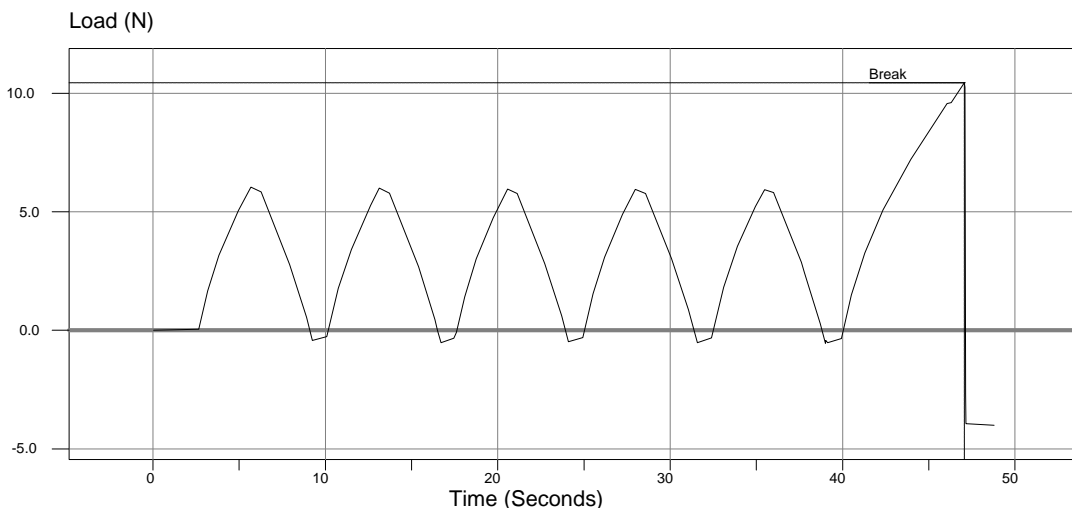
```

If broken then
  Result "Break Load", LoadAtBreak
  Result "Break Extension", ExtnAtBreak
End If
    
```

Notes for Example 4.29

This test is basically a combination of two previous tests.

The sample is cycled between the 2 stages until either 50 cycles have been completed or the user clicks on the Advance button. The break detector is switched on after the cycling stages have been performed then the last stage will move the crosshead to break the sample.



Primary Script

'Example 4.29 - Combined Cycling and Pull to Break

'Ensure break detector is selected on the Test Parameters screen

Option Explicit

'Traps spelling mistakes of variable names

Dim count

'Declare variable to be used when using option explicit

AdvanceOn

'enable the advance button

For count = 1 to 50

' alter the 50 to the required number of cycles

 progress "Running Cycle No " & count

' display cycle number on the status bar

 Stage [50mm], [1000mm/min]

' Alter limit and speed as required

 Stage [0mm], [1000mm/min]

' Alter limit and speed as required

 if advancing then exit for

' end the cycling if the advance button is pressed

Next

AdvanceOff

BreakOn

'activate break detector

progress "Pulling to Break"

'display message on the status bar

Stage [50kN], [1000mm/min]

'Use a load limit of the loadcell value

Result "No of Cycles", count - 1

'Number of COMPLETED cycles

'Break results are obtained in the secondary script

4.30 Combined Cycling and Creep Test

A sample is stretched by 50mm then released for up to 50 cycles then it is loaded to 50N and this load maintained for a 10 minute time period. The increase in extension (sample creep) over this time period is required. The cyclic part of the test can be ended early by clicking on the Advance Button and this will be indicated by the Number of Cycles Result.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not expected
Auto Zero	Off
Auto Return	On
Markers	0

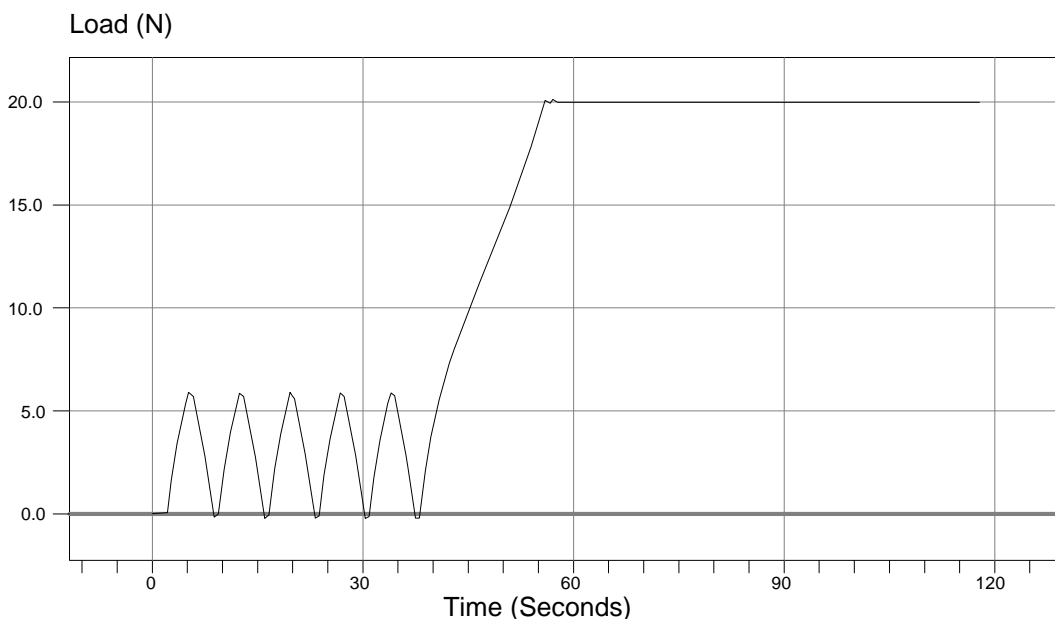
The Primary Script is shown on the next page
Secondary Script

Not used

Notes for Example 4.30

This test is basically a combination of two previous tests.

The sample is cycled between the 2 stages until either 50 cycles have been completed or the user clicks on the Advance button. The extension at the 50N loading is then measured and the load maintained for either 10 minutes or until the user clicks on the End button. The extension is measured again and the change in extension reported as the sample creep.



Primary Script

'Example 4.30 - Combined Cycling and Constant Loading

'Ensure break detector is switched off

Option Explicit

Dim count

Dim refext

Dim reftime

AdvanceOn

'enable the advance button

For count = 1 to 50

' alter the 50 to the required number of cycles

 progress "Running Cycle No " & count

' display cycle number on the status bar

 Stage [50mm], [100mm/min]

' Alter limit and speed as required

 Stage [0mm], [100mm/min]

' Alter limit and speed as required

 if advancing then exit for

' end the cycling if the advance button is pressed

Next

AdvanceOff

progress "Constant Loading"

'display message on the status bar

Stage [20N], [100mm/min]

Set refext = Extn

Set reftime = Testtime

EndOn

'enable the end button

Stagehold [50N], [100mm/min], [10min]

'alter load, speed and time as required

Result "Sample Creep", [Extn - refext]

Result "Hold Time", [Testtime - reftime]

Result "No of Cycles", count - 1

4.31 Combined Cycling and Relaxation Test

A sample is stretched by 50mm then released for up to 50 cycles then it is stretched to 50mm and the crosshead maintained at this position for a 10 minute time period. The decrease in load (sample relaxation) over this time period is required. The cyclic part of the test can be ended early by clicking on the Advance Button and this will be indicated by the Number of Cycles Result.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not expected
Auto Zero	Off
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

Not used

Notes for Example 4.31

This test is basically a combination of two previous tests.

The sample is cycled between the 2 stages until either 50 cycles have been completed or the user clicks on the Advance button.

The load at 50mm is then measured and the load maintained for either 10 minutes or until the user clicks on the End button. The load is measured again and the change in load reported as the sample relaxation.

Primary Script

'Example 4.31 - Combined Cycling and Relaxation

'Ensure break detector is switched off

Option Explicit

Dim count

Dim reflow

Dim reftime

AdvanceOn

'enable the advance button

For count = 1 to 50

' alter the 50 to the required number of cycles

 progress "Running Cycle No " & count

' display cycle number on the status bar

 Stage [50mm], [100mm/min]

' Alter limit and speed as required

 Stage [0mm], [100mm/min]

' Alter limit and speed as required

 if advancing then exit for

' end the cycling if the advance button is pressed

Next

AdvanceOff

progress "Sample Relaxation"

'display message on the status bar

Stage [50mm], [100mm/min]

Set reflow = Load

Set reftime = Testtime

EndOn

'enable the end button

Stagehold [50mm], [100mm/min], [10min]

'alter extension, speed and time as required

Result "Sample Relaxation", [refload - Load]

Result "Hold Time", [Testtime - reftime]

Result "No of Cycles", count - 1

4.32 Foam Test 1

The thickness of a block of foam is automatically measured by the machine then it is compressed by 40% and the crosshead held at this position for 1 minute. The force is then removed and the thickness re-measured. The foam is allowed to expand for 1 minute then the thickness measured again. The results required are the initial thickness, the thickness after the force has been removed and the thickness after the recovery period..

Test Parameters

Direction	Compression
Preload	1.00 N
Height	Auto-Measure
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

Not used

Notes for Example 4.32

This test uses the Auto Height Feature which requires the Datum position to be defined. The test preload force and the datum preload force are both set to **1N** and the same contact force is used in the 2 stages which re-measures the height of the foam during the test.

The first stage compresses the sample by **40%** but as there is no % unit for a stage limit, this has to be calculated using the formula **Height * 0.4**. This position is maintained for 1 minute then the force is reduced to the preload force value and the extension measured. The sample height is the difference between the original height (stored in the variable Height) and the measured extension, i.e. **Height - Extn**. The crosshead is moved to a negative position (above the sample preload point) by a limit of **-10mm** and the sample is allowed to expand for 1 minute.

The crosshead is moved down again to re-apply the preload force value and the extension measured. The sample height is again the difference between the original height (stored in the variable Height) and the measured extension, i.e. **Height - Extn**.

Primary Script

'Example 4.32 - Foam Test 1

'THE ZERO BUTTON MUST NOT BE PRESSED AFTER STARTING THIS TEST

'Select Compression

'Ensure break detector is switched off

'SET PRELOAD AND AUTO HEIGHT PRELOAD TO 1N

'ENSURE AUTO ZERO IS SWITCHED OFF

'Alter values to suit the sample

Result "Initial Height", Height

'Auto-Height measurement using the Datum position

Progress "Compressing Sample"

Stagehold [height * 0.4], [100mm/min], [1min]

'compress to 40% of height measured from datum

'alter % and hold time as required

Progress "Removing Force from Sample"

Stage [1N], [100mm/min]

'return to the value of the preload force

Result "Height after Compression", [Height - Extn]

'measure extension immediately the force is removed

'note the square brackets

Progress "Allowing Sample to Expand"

Stagehold [-10mm], [100mm/min], [1min]

'lift top compression plate 10mm above sample

'alter hold time as required

Progress "Compressing Sample"

Stage [1N], [100mm/min]

'return to the value of the preload force

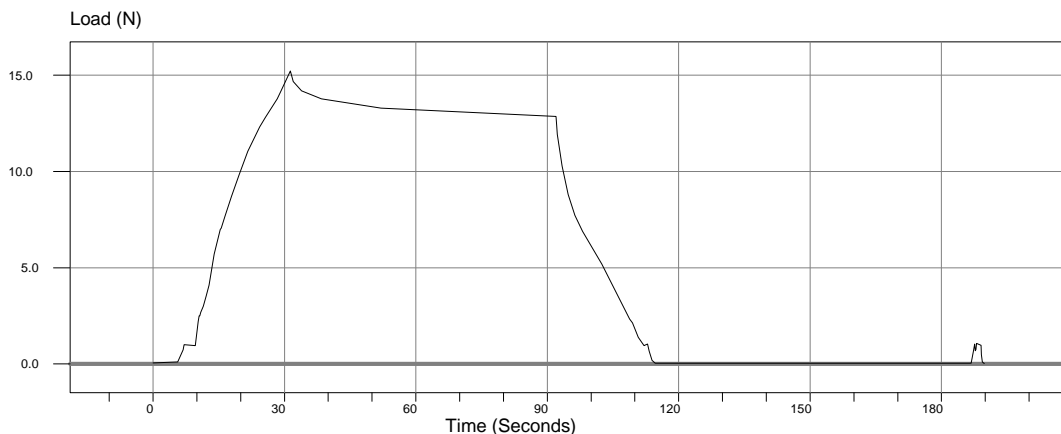
Result "Height after Relaxation", [Height - Extn]

'measure extension after sample has expanded

'note the square brackets

Testing the Samples

- 1 Attach the compression plates to the machine.
- 2 Move the crosshead using the jog keys on either the machines console or the software console until the distance between the plates is sufficient to fit the sample without force.
- 3 **ZERO** the machine using the zero button on either the machines console or the software console.
- 4 Do **NOT** click on the **START TEST** Icon or press the **F5** key to display the pretest dialogue box. If this has been displayed by accident, click on the **ABORT** button to close it and return to the graph screen.
- 5 Manually move the crosshead down, using the jog keys on either the machines console or the software console, until the grip separation is approx. 5mm. **DO NOT PRESS THE ZERO BUTTON ON THE MACHINE OR THE SOFTWARE CONSOLE.**
- 6 Click on the **START TEST** Icon or press the **F5** key to display the pretest dialogue box. Enter the required details then click on the OK button. The crosshead will move downwards at the DATUM SPEED to apply the DATUM force.
- 7 The crosshead will automatically return at maximum speed to the previously defined ZERO position. **NOTE THAT THE ZERO BUTTON MUST NOT BE PRESSED DURING THE TESTING OR THE DATUM POSITION WILL BE LOST.**
- 8 Fit the sample then click on the OK button to start the test.
- 9 The crosshead will move downwards at the PRELOAD SPEED until the PRELOAD force is applied.
- 10 The main part of the test will be performed and the results stored in the batch table.
- 11 If another sample is to be tested, fit it between the plates then click on the **START TEST** Icon or press the **F5** key to display the pretest dialogue box. This box will now indicate that a Datum position is known so enter the required details then accept this position by clicking on the OK button.



4.33 Foam Test 2

The thickness of a block of foam is automatically measured by the machine then it is compressed by 75%. The force is removed then it is compressed by 75% again. The force is removed and the foam is allowed to expand for 6 minutes. The crosshead is moved downwards to apply a force of 4.5N then the thickness of the foam measured. The foam is compressed to give an indentation of 25% of it's original thickness and the load is measured. The foam is compressed to give an indentation of 65% of it's original thickness and the load is measured. The results required are the forces at the 25% and 65% indentations.

Test Parameters

Direction	Compression
Preload	4.50 N
Height	Auto-Measure
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

The Primary Script is shown on the next page
Secondary Script

Not used

Notes for Example 4.33

This test is based upon the ASTM D3574 part B1 which specified a contact force of 4.5N, a preconditioning speed of 200mm/min and a test speed of 50mm/min.

This test uses the Auto Height Feature which requires the Datum position to be defined using a test preload force and datum preload force of **4.5N**. The first stage compresses the sample by **70%** which is calculated using the formula **Height * 0.7**. The force is removed then the sample compressed by 70% again. The force is removed and the sample allowed to expand by using a stagehold command with a limit of -10mm and a hold time of 6 minutes which lifts the top plate 10mm above the sample.

The next 2 stages compress the sample by 25% and 65% and the force is measured at each position. The crosshead position is defined using 2 SetupStages then moved using 2 Runstages to eliminate the slight delay (and load drop) that would occur if 2 stage command were used.

Primary Script

```
'Example 4.33 - Foam Test 2

'THE ZERO BUTTON MUST NOT BE PRESSED AFTER STARTING THIS TEST

'Select Compression
'Ensure break detector is switched off
'SET PRELOAD AND AUTO HEIGHT PRELOAD TO 4.5N
'ENSURE AUTO ZERO IS SWITCHED OFF
'Alter values to suit the sample

'This test is based upon ASTM D3574 part B1

Progress "Conditioning Sample"

Stage [Height * 0.7], [200mm/min]
'compress to 70% of height measured from datum
'alter % as required
Stage [0mm], [200mm/min]
Stage [Height * 0.7], [200mm/min]

Progress "Allowing Sample to Expand"
StageHold [-10mm], [200mm/min], [6min]

SetupStage [Height * 0.25], [50mm/min]
SetupStage [Height * 0.65], [50mm/min]
'define the next 2 stages to eliminate the slight delay when using 2 Stages

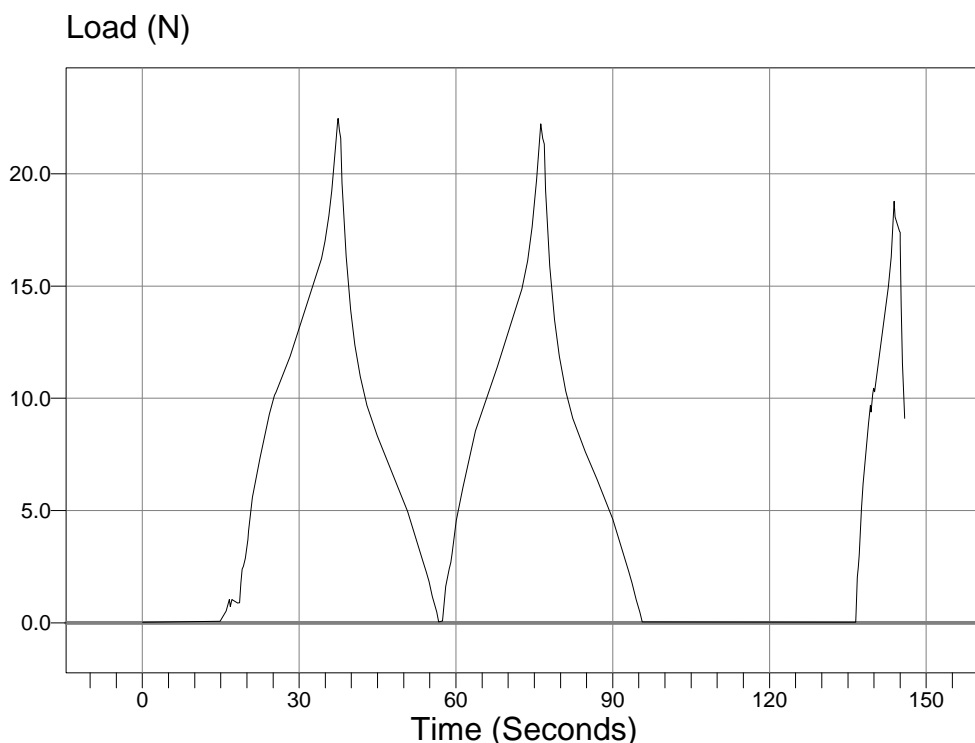
Progress "Compressing to 25%"
Runstage
Result "25% IFD", load

Progress "Compressing to 65%"
Runstage
Result "65% IFD", load
```

Testing the Samples

- 1 Attach the compression plates to the machine.
- 2 Move the crosshead using the jog keys on either the machines console or the software console until the distance between the plates is sufficient to fit the sample without force.
- 3 **ZERO** the machine using the zero button on either the machines console or the software console.

- 4 Do **NOT** click on the **START TEST** Icon or press the **F5** key to display the pretest dialogue box. If this has been displayed by accident, click on the **ABORT** button to close it and return to the graph screen.
- 5 Manually move the crosshead down, using the jog keys on either the machines console or the software console, until the grip separation is approx. 5mm. **DO NOT PRESS THE ZERO BUTTON ON THE MACHINE OR THE SOFTWARE CONSOLE.**
- 6 Click on the **START TEST** Icon or press the **F5** key to display the pretest dialogue box. Enter the required details then click on the OK button. The crosshead will move downwards at the DATUM SPEED to apply the DATUM force.
- 7 The crosshead will automatically return at maximum speed to the previously defined ZERO position. **NOTE THAT THE ZERO BUTTON MUST NOT BE PRESSED DURING THE TESTING OR THE DATUM POSITION WILL BE LOST.**
- 8 Fit the sample then click on the OK button to start the test.
- 9 The crosshead will move downwards at the PRELOAD SPEED until the PRELOAD force is applied.
- 10 The main part of the test will be performed and the results stored in the batch table.
- 11 If another sample is to be tested, fit it between the plates then click on the **START TEST** Icon or press the **F5** key to display the pretest dialogue box. This box will now indicate that a Datum position is known so enter the required details then accept this position by clicking on the OK button.



4.34 Foam Test 3

The thickness of a block of foam is automatically measured by the machine then it is compressed by applying a force of 330N. The force is removed then it is loaded to 330N again. The force is removed and the foam is allowed to expand for 6 minutes. The foam is compressed by applying forces of 4.5N, 110N and 220N and the thickness is measured at each loading.. The results required are the foam thickness at the 4.5N, 110N and 220N forces.

Test Parameters

Direction	Compression
Preload	4.50 N
Height	Auto-Measure
Area	Not Applicable
Break	Not Expected
Auto Zero	Off
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

Not used

Notes for Example 4.34

This test is based upon the ASTM D3574 part B2 which specifies a contact force of 4.5N, a preconditioning speed of 200mm/min and a test speed of 50mm/min.

This test uses the Auto Height Feature which requires the Datum position to be defined. The test preload force and the datum preload force are both set to **4.5N**. The first three stages apply a force of 300N, remove the force then re-apply it again. The force is removed then the sample allowed to expand by using a stagehold command with a limit of **-10mm** and a hold time of 6 minutes which lifts the top plate 10mm above the sample.

The next 3 stages touch the sample then apply forces of 4.5N, 110N and 220N and the sample height measured at each load. The loads are defined using 3 SetupStages then moved using 3 RunStages to eliminate the slight delay (and load drop) that would occur if 3 Stage commands were used. The sample height at each load is the difference between the original height (stored in the variable Height) and the measured extension, i.e. **Height - Extn**.

Primary Script

```
'Example 4.34 - Foam Test 3

'THE ZERO BUTTON MUST NOT BE PRESSED AFTER STARTING THIS TEST

'Select Compression
'Ensure break detector is switched off
'SET PRELOAD AND AUTO HEIGHT PRELOAD TO 4.5N
'ENSURE AUTO ZERO IS SWITCHED OFF
'Alter values to suit the sample

'This test is based upon ASTM D3574 part B2

Progress "Conditioning Sample"

Stage [330N], [200mm/min]
'apply a force of 330N
Stage [0mm], [200mm/min]
Stage [330N], [200mm/min]

Progress "Allowing Sample to Expand"
StageHold [-10mm], [200mm/min], [6min]

SetupStage [4.5N], [200mm/min]
SetupStage [110N], [50mm/min]
SetupStage [220N], [50mm/min]
'define the next 3 stages to eliminate the slight delay when using 3 Stages

Progress "Finding Height"
Runstage
Result "Thickness", [Height - Extn]
'Find new height of foam

Progress "Compressing to 110N"
Runstage
Result "Thickness at 110N", [Height - Extn]

Progress "Compressing to 220N"
Runstage
Result "Thickness at 220N", [Height - Extn]
```

Testing the Samples

- 1 Attach the compression plates to the machine.
- 2 Move the crosshead using the jog keys on either the machines console or the software console until the distance between the plates is sufficient to fit the sample without force.
- 3 **ZERO** the machine using the zero button on either the machines console or the software console.
- 4 Do **NOT** click on the **START TEST** Icon or press the **F5** key to display the pretest dialogue box. If this has been displayed by accident, click on the **ABORT** button to close it and return to the graph screen.
- 5 Manually move the crosshead down, using the jog keys on either the machines console or the software console, until the grip separation is approx. 5mm. **DO NOT PRESS THE ZERO BUTTON ON THE MACHINE OR THE SOFTWARE CONSOLE.**
- 6 Click on the **START TEST** Icon or press the **F5** key to display the pretest dialogue box. Enter the required details then click on the OK button. The crosshead will move downwards at the DATUM SPEED to apply the DATUM force.
- 7 The crosshead will automatically return at maximum speed to the previously defined ZERO position. **NOTE THAT THE ZERO BUTTON MUST NOT BE PRESSED DURING THE TESTING OR THE DATUM POSITION WILL BE LOST.**
- 8 Fit the sample then click on the OK button to start the test.
- 9 The crosshead will move downwards at the PRELOAD SPEED until the PRELOAD force is applied.
- 10 The main part of the test will be performed and the results stored in the batch table.
- 11 If another sample is to be tested, fit it between the plates then click on the **START TEST** Icon or press the **F5** key to display the pretest dialogue box. This box will now indicate that a Datum position is known so enter the required details then accept this position by clicking on the OK button.

4.35 Bottle Test 1

A nozzle is to be pushed onto the top of detergent bottle and the "Capping Force" measured. After the nozzle has been clicked onto the bottle, the complete bottle is to be crushed by 4mm and the "Top Load" (force at 4mm compression) measured. The results required are the Capping Force and the Top Load.

Test Parameters

Direction	Compression
Preload	1 N
Height	Not Applicable
Area	Not Applicable
Break	Load drops to 95 %
Auto Zero	On
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

Not used

Notes for Example 4.35

This test uses the Manual Break Detector which is set to detect a "Break" when the load falls to 95% of the maximum load. The nozzle is placed loosely on top of the bottle and the assembly placed centrally on the lower compression plate. The plate separation is such that the assembly can be easily inserted without excessive clearance. The top plate moves down at the preload speed of 50mm/min until the contact force of 1N is applied.

The first stage continues to move the top plate down and the break detector waits for the load to fall to indicate that the nozzle has snapped onto the bottle. The first stage uses a limit of 10mm which will stop the test if the load does not drop for any reason.

When the load drops, the break detector will trigger and the first stage will end. The break detector sets the Boolean variable called "Broken" to TRUE and will cause ALL Stage or Run commands to be ignored. The IF statement is actioned and the Break Detector is switched off to allow the next stage to be performed. The maximum force during the first stage is reported with the title "Capping Force" then the second stage moves the crosshead 4mm lower than the position at the END of the first stage. The force at the 4mm crush is reported with the title "Top Load".

If the nozzle does not snap onto the bottle, the load will not drop during the first stage and the stage will end after the plates have moved down 10mm from the contact position. The variable "Broken" will still be FALSE so the IF statement will NOT be actioned and NO results will be reported.

Primary Script

```
'Example 4.35 - Bottle Test 1

'ENSURE COMPRESSION IS SELECTED
'Use 1N preload with 50mm/min to start the graph
'Ensure that the Break Detector is set to Load drops to 95% and 10N start load
'Alter values to suit the sample

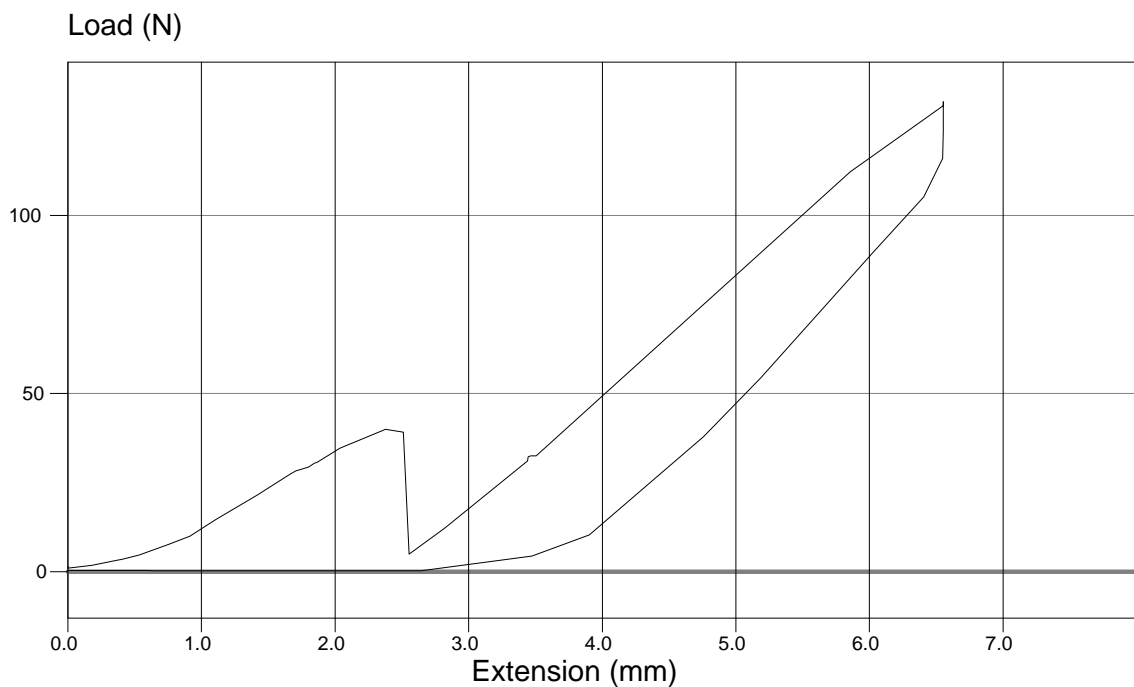
Progress "Pushing Cap onto Bottle"
'display message on status bar

Stage [10mm], [50mm/min]
'Alter limit and speed as required
'limit is used a safety limit is snap is not detected

If Broken then
  BreakOff
  Result "Capping Force", LoadAtStageMax
  Progress "Crushing Bottle"

  Stage [Extn + [4mm] ], [100mm/min]
  ' move crosshead down by 4mm after snap is detected
  ' alter limit and speed as required

  Result "Top Load", Load
End If
```



4.36 Bottle Test 2

A nozzle is to be pushed onto the top of detergent bottle and the "Capping Force" measured. After the nozzle has been clicked onto the bottle, the crosshead is lifted and the "Pull Off Force" measured. The results required are the Capping Force and the Pull Off Force.

Test Parameters

Direction	Compression
Preload	1 N
Height	Not Applicable
Area	Not Applicable
Break	Load drops to 95 %
Auto Zero	On
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

Not used

Notes for Example 4.36

This test uses the Manual Break Detector which is set to detect a "Break" when the load falls to 95% of the maximum load. The nozzle is placed loosely on top of the bottle and the assembly placed centrally on the lower compression plate. The plate separation is such that the assembly can be easily inserted without excessive clearance. The top plate moves down at the preload speed of 50mm/min until the contact force of 1N is applied.

The first stage continues to move the top plate down and the break detector waits for the load to fall to indicate that the nozzle has snapped onto the bottle. The first stage uses a limit of 10mm which will limit the crosshead travel if the load does not drop for any reason. When the load drops, the break detector will trigger and the first stage will end. The break detector sets the Boolean variable called "Broken" to TRUE and will cause ALL Stage or Run commands to be ignored so the Break Detector is switched off by the BreakOff command to allow the next stage to be performed. The maximum force during the first stage is reported with the title "Capping Force" then the second stage returns the crosshead back to 0N to remove the load.

Note that the load MUST be removed BEFORE the Break Detector is re-enabled otherwise the Break Detector will trigger as the load falls back to zero.

The Break Detector is re-enabled then the last stage pulls the nozzle back off the bottle. When the load drops, the break detector will trigger, the last stage will end and the maximum negative force during the stage is reported with the title "Pull Off Force".

Primary Script

```
'Example 4.36 - Bottle Test 2

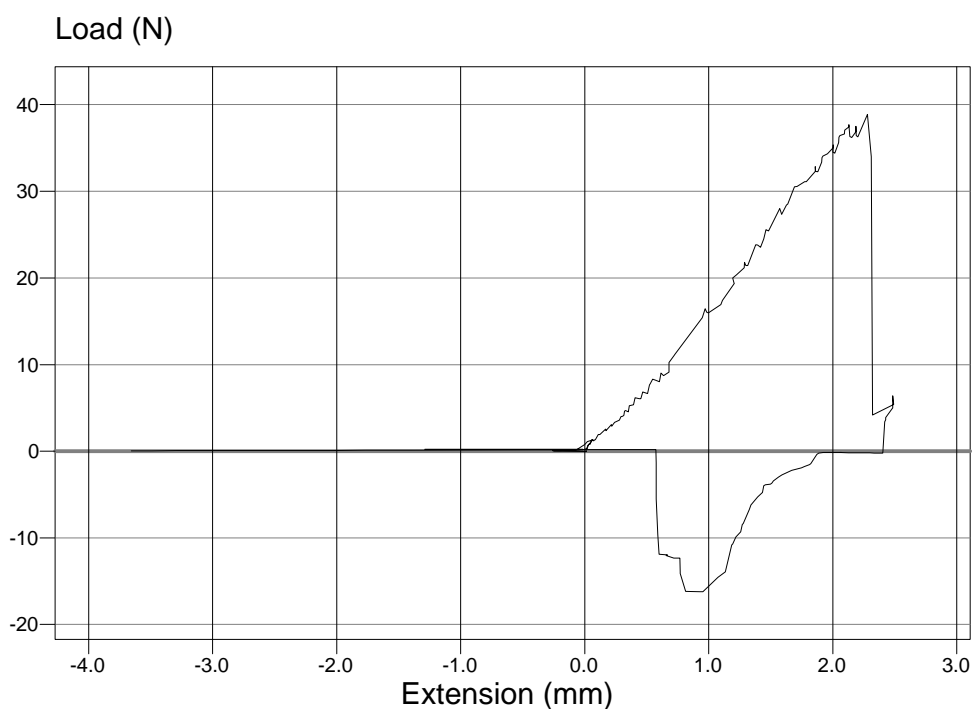
'ENSURE COMPRESSION IS SELECTED
'Use 1N preload with 10mm/min to start the graph
'Ensure that the Break Detector is set to Load drops to 95% and 10N start load
'Alter values to suit the sample

Progress "Pushing Cap onto Bottle"
'display message on status bar

SetupStage [10mm], [10mm/min]
Setupstage [0N], [10mm/min]
SetupStage [-10mm], [10mm/min]
'Alter limits and speeds as required

RunStage
Result "Capping Force", LoadAtStageMax
Progress "Pulling Cap off Bottle"
BreakOff
'Disable Break Detector when removing force
RunStage
'remove load from sample by using a stage limit of 0N
Breakon
'Enable Break Detector ready for negative load drop
RunStage

Result "Pull Off Force", LoadAtStageMin
```



4.37 Pull to Break with Real Time Event Marker

A sample is to be loaded until it breaks and the graph marked at the point where some physical change took place which could not be detected by measuring the load, e.g. a change in colour occurs or a transparent sample becomes crazed etc. The user observes the sample and when the change occurs, it is marked by clicking on the advance button. The results required are the Event Force , the Maximum Force and the Force at Break.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Load drops quickly
Auto Zero	On
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

'Example 4.37 - Pull to Break with Real Time Event Marker

If Broken then result "Break Load", LoadAtBreak

Notes for Example 4.37

This test uses an event marker which is an internal feature of Ondio designed for use by Lloyd Instruments Software Engineers so is a little more complicated than other Ondio commands. This feature will be added as an Ondio command in future release.

The test is defined using 2 identical SetupStage commands. The first stage is run, using the first RunStage command and the crosshead will move upwards to break the sample because the stage limit is a high load value. If the user does not click on the advance button, the sample will break and the stage will end. The variable "Advancing" will not be set, so the IF statement will not be actioned. The result called "Event" will report FALSE because no event was marked. The variable "Broken" will be set and the last RunStage will not be performed so the test will end.

If the user needs to mark an event, this is marked by clicking on the advance button which causes the first stage to end. The advance button sets the Boolean variable called "advancing" to TRUE and will cause ALL Stage or Run commands to be ignored. The IF statement is actioned and the load at the END of the first stage is reported with the title "Load at Event". The result called "Event" will report TRUE because an event was marked. The advanceoff command allows the next RunStage to perform the second SetupStage and the test will end when the sample breaks.

Primary Script

'Example 4.37 - Pull to Break with Real Time Marker

'Ensure break detector is selected on the Test Parameters screen

AdvanceOn

'enable the advance button

SetupStage [5kN], [100mm/min]

SetupStage [5kN], [100mm/min]

'Use a load limit of the loadcell value

progress "Click on Advance to Mark the Event"

'display message on the status bar

RunStage

if advancing then

 Result "Load at Event", Load

 Foundation.Graph.InsertEvent "Event", Foundation.StageEndTime

' places an event cross on the graph

 Progress "Event Marked"

' display message on the status bar

end if

Result "Event", advancing

'indicates "True" if an event was marked by the advance button

 AdvanceOff

'allows the runstage to be performed if advance button was pressed

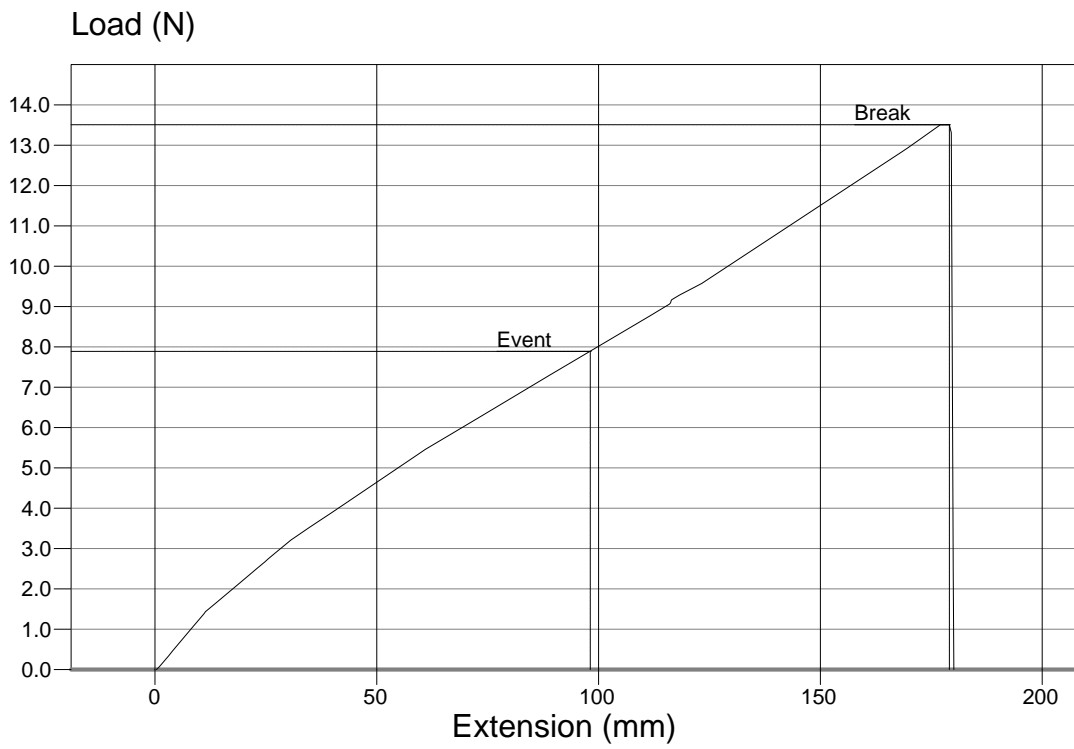
 RunStage

'only used if the event is marked otherwise it is ignored by the sample breaking

Result "Maximum Load", LoadAtMax

'Break result is obtained in the secondary script

Graph for Example 4.37



Advanced pressed during the test

4.38 Cycling with Real Time Event Marker

A sample is to be loaded and unloaded a fixed number of cycles and the graph marked at the point where some physical change took place which could not be detected by measuring the load, e.g. a change in colour occurs or a transparent sample becomes crazed etc. The user observes the sample and when the change occurs, it is marked by clicking on the advance button. The results required are the Event Force and the Cycle when the event occurred.

Test Parameters

Direction	Tension
Preload	None
Height	Not Applicable
Area	Not Applicable
Break	Not expected
Auto Zero	On
Auto Return	On
Markers	0

**The Primary Script is shown on the next page
Secondary Script**

Not Used

Notes for Example 4.38

This test uses an event marker which is an internal feature of Ondio designed for use by Lloyd Instruments Software Engineers so is a little more complicated than other Ondio commands. This feature will be added as an Ondio command in future release.

The test is basically a 5 cycle test using a FOR - NEXT loop. If the user does not click on the advance button, the variable "advancing" will be FALSE and the IF statements will not be actioned. The test will end after 5 cycles and the result called "Event" will report FALSE because no event was marked. The results of "Event Load" and "Event Cycle" will not report any values.

If the event is marked when the load is being applied, the advance button sets the Boolean variable called "advancing" to TRUE and the stage will end. The first IF statement will be actioned and the load at the END of the first stage will be reported with the title "Event Load". The cycle number will be reported with the title "Event Cycle" and the result called "Event" will report TRUE because an event was marked. The advanceoff command allows the next Stage to continue loading the sample to the original first limit.

If the event is marked when the load is being removed, the advance button sets the Boolean variable called "advancing" to TRUE and the stage will end. The first IF statement will be actioned and the load at the END of the first stage will be reported with the title "Event Load". The cycle number will be reported with the title "Event Cycle" and the result called "Event" will report TRUE because an event was marked. The advanceoff command allows the next Stage to continue unloading the sample to the original second limit.

Primary Script

```
'Example 4.38 - Cycling with Real Time Marker

'Ensure that the break detector is switched off

Option Explicit
Dim count
Dim button
button = false

AdvanceOn

Progress "Click on Advance to Mark the Event"

For count = 1 to 5
'   alter the 5 to the required number of cycles

    Stage [50mm], [100mm/min]
'   Alter limit and speed as required

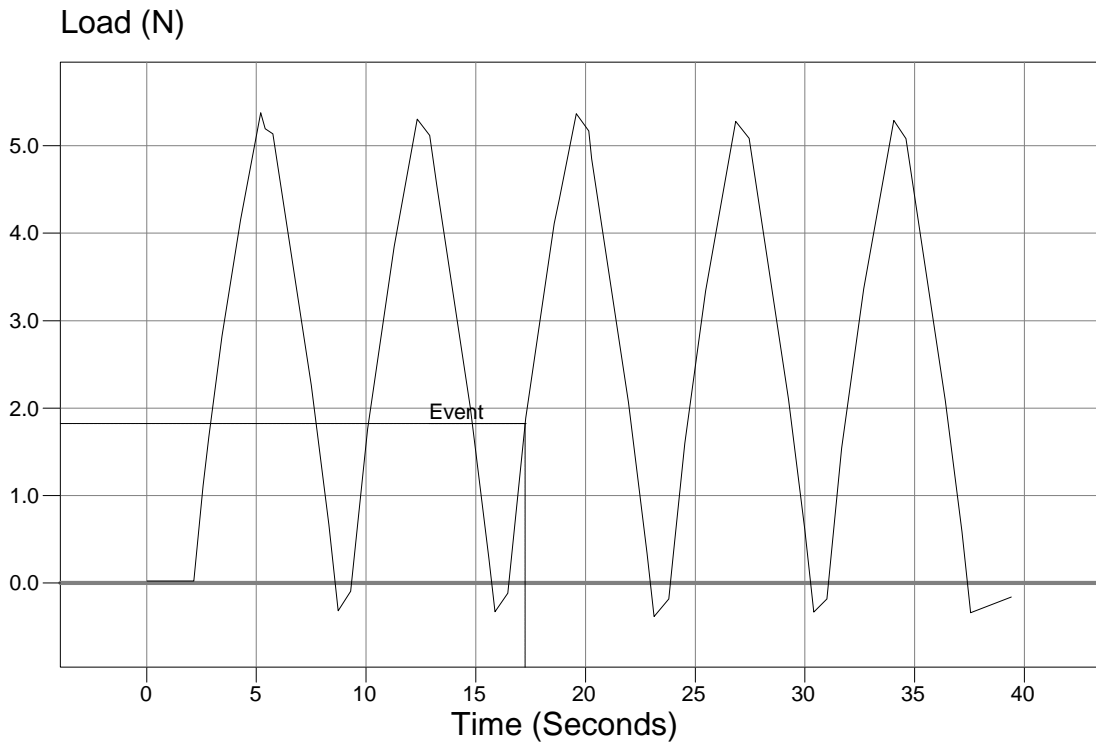
    If advancing then
        Result "Event Load", Load
        Result "Event Cycle", count
        button = true
        Foundation.Graph.Insertevent "Event", Foundation.StageEndTime
'   places an event cross on the graph
        Progress "Event Marked"
        AdvanceOff
        Stage [50mm], [100mm/min]
    End If

    Stage [0mm], [100mm/min]
'   Alter limit and speed as required

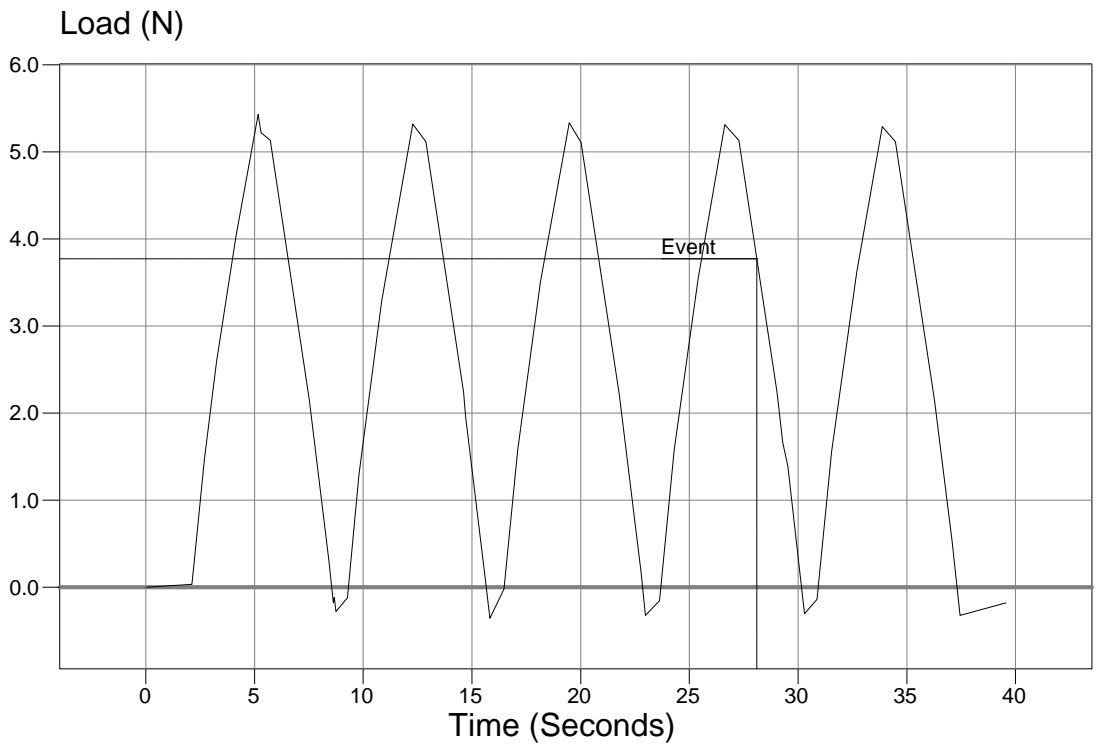
    If advancing then
        Result "Event Load", Load
        Result "Event Cycle", count
        button = true
        Foundation.Graph.Insertevent "Event", Foundation.StageEndTime
'   places an event cross on the graph
        Progress "Event Marked"
        AdvanceOff
        Stage [0mm], [100mm/min]
    End If
Next

Result "Event", button
```

Graphs for Example 4.38



Advanced pressed when load is increasing



Advanced pressed when load is decreasing

4.39 Pausing the Test for Special Purposes

Some tests may require the crosshead to remain at a fixed position to wait for a condition to occur, e.g. to wait for the thermal chamber to stabilise, before testing the sample. This can be achieved in several ways and 3 ways are listed below:-

Using a StageHold Command with a Fixed Time Period

Use a StageHold command with a suitable extension limit and time period.

STAGEHOLD [0mm], [10mm/min], [10min]	Wait for 10 minutes
STAGE [10mm], [100mm/min]	Test the sample
Result "Load at 10mm", Load	Measure result

Using a StageHold Command with the End Button

Use a long time period and activate the END button using the EndOn command.

EndOn	Activate the End button
PROGRESS "Click the End Button to Continue"	Message for User
STAGEHOLD [0mm], [10mm/min], [1hours]	Wait for 1 hour
EndOff	Allow next stage to be performed
STAGE [10mm], [100mm/min]	Test the sample
Result "Load at 10mm", Load	Measure result

When the End button is clicked, the StageHold is ended and the test stage is performed. Note that after the End button has been clicked, ALL subsequent Stage and RunStage commands will be ignored until an EndOff command is used. The End button is only used to terminate the StageHold command so an EndOff command is required on the following line.

However, it may not be obvious to the user that the test will only continue when the END button is clicked, even though a message is displayed on the status bar.

Using the StopMachine Command with a Dialogue Box

The **STOPMACHINE** command turns off the motor drive and should **ONLY** be used when there is no load on the sample so is not recommended for relaxation tests etc. The command disables the machine BUT the script is still actioned and the machine will restart when it receives the next STAGE or RUNSTAGE command.

The script below will NOT pause the machine because the Stage command is actioned immediately after the STOPMACHINE command.

STOPMACHINE	Disable Motor Drive
STAGE [10mm], [100mm/min]	Test the sample
Result "Load at 10mm", Load	Measure result

To allow the STOPMACHINE command to pause the machine as required, the script must ALSO be paused at the required point and this is achieved by creating a VBScript "OK" Dialogue Box as shown below:-

STOPMACHINE	Disable Motor Drive
MSGBOX "Press OK to Continue the Test"	Create Dialogue Box
STAGE [10mm], [100mm/min]	Test the sample
Result "Load at 10mm", Load	Measure result

The screen below shows a test which compresses a sample 5 times then allows the user to remove the sample to weigh it. The sample is then refitted to the compression plates then cycled again. The VB Script Dialogue Box displays an "OK" button together with the text typed within quotes " ". The test is still in progress as shown by the rotation of the Lloyd Logo in the top left of the screen.

