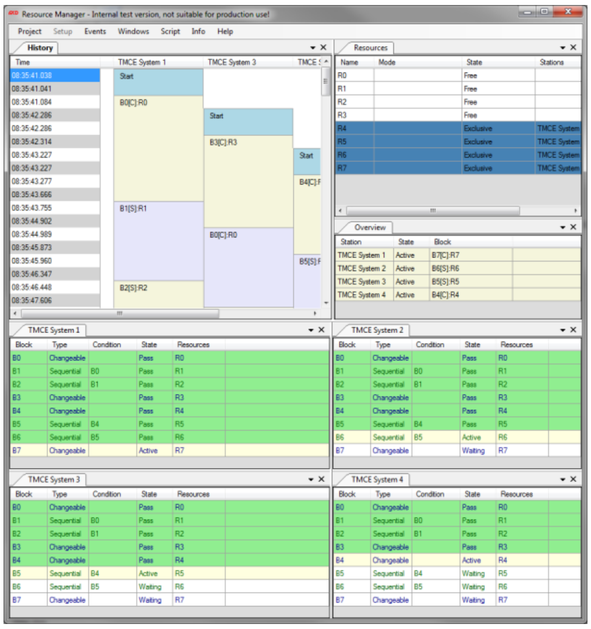
ELEKTRONIK GMBH

GET IN **touch** WITH SENSITIVE TESTING

Manual

Toolmonitor ResourceManager


Resource Manager - Internal test version, not suitable for production use!

Project Setup Events Windows Script Info Help

History

Time	TMCE System 1	TMCE System 3	TMCE S
08:35:41.038	Start		
08:35:41.041			
08:35:41.084	B0[C]:R0		
08:35:42.286		Start	
08:35:42.286			
08:35:42.314		B3[C]:R3	
08:35:43.227			Start
08:35:43.227			
08:35:43.277			B4[C]:F
08:35:43.666			
08:35:43.755	B1[S]:R1		
08:35:44.902			
08:35:44.989		B0[C]:R0	
08:35:45.873			
08:35:45.960			B5[S]:F
08:35:46.347			
08:35:46.448	B2[S]:R2		
08:35:47.606			

Resources

Name	Mode	State	Stations
R0		Free	
R1		Free	
R2		Free	
R3		Free	
R4		Exclusive	TMCE System
R5		Exclusive	TMCE System
R6		Exclusive	TMCE System
R7		Exclusive	TMCE System

Overview

Station	State	Block
TMCE System 1	Active	B7[C]:R7
TMCE System 2	Active	B6[S]:R6
TMCE System 3	Active	B5[S]:R5
TMCE System 4	Active	B4[C]:R4

TMCE System 1

Block	Type	Condition	State	Resources
B0	Changeable		Pass	R0
B1	Sequential	B0	Pass	R1
B2	Sequential	B1	Pass	R2
B3	Changeable		Pass	R3
B4	Changeable		Pass	R4
B5	Sequential	B4	Pass	R5
B6	Sequential	B5	Pass	R6
B7	Changeable		Active	R7

TMCE System 2

Block	Type	Condition	State	Resources
B0	Changeable		Pass	R0
B1	Sequential	B0	Pass	R1
B2	Sequential	B1	Pass	R2
B3	Changeable		Pass	R3
B4	Changeable		Pass	R4
B5	Sequential	B4	Pass	R5
B6	Sequential	B5	Active	R6
B7	Changeable		Waiting	R7

TMCE System 3

Block	Type	Condition	State	Resources
B0	Changeable		Pass	R0
B1	Sequential	B0	Pass	R1
B2	Sequential	B1	Pass	R2
B3	Changeable		Pass	R3
B4	Changeable		Pass	R4
B5	Sequential	B4	Active	R5
B6	Sequential	B5	Waiting	R6
B7	Changeable		Waiting	R7

TMCE System 4

Block	Type	Condition	State	Resources
B0	Changeable		Pass	R0
B1	Sequential	B0	Pass	R1
B2	Sequential	B1	Pass	R2
B3	Changeable		Pass	R3
B4	Changeable		Active	R4
B5	Sequential	B4	Waiting	R5
B6	Sequential	B5	Waiting	R6
B7	Changeable		Waiting	R7

Softline

Modline

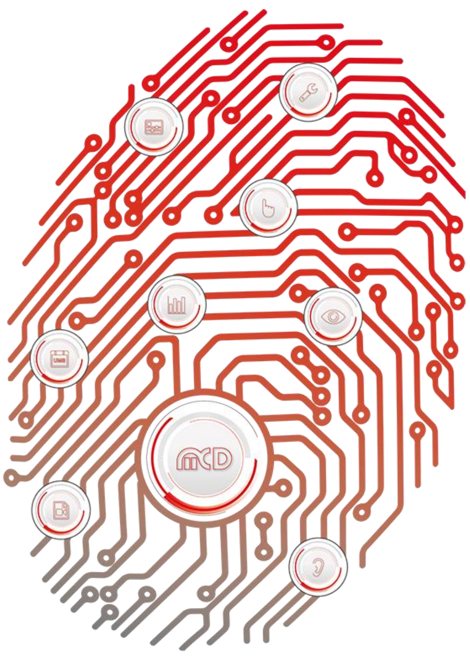
Conline

Boardline

Avidline

Pixline

Application



MCD Elektronik GmbH
Hoheneichstr. 52
75217 Birkenfeld
Tel. +49 (0) 72 31/78 405-0
Fax +49 (0) 72 31/78 405-10
info@mcd-elektronik.de
www.mcd-elektronik.com
HQ: Birkenfeld
Managing CEO: Bruno Hörter
Register Court Mannheim
HRB 505692

Template version: 5.4 / 2018-04-09
V1.2 2018-05-03 (TS)

Table of Contents

1. General

In combination with TestManager, ResourceManager serves to control the process control system for test systems, through which multiple devices will be tested simultaneously and asynchronously while using the existing hardware for the respective measurements in a shared manner.

Working with ResourceManager provides the following capabilities:

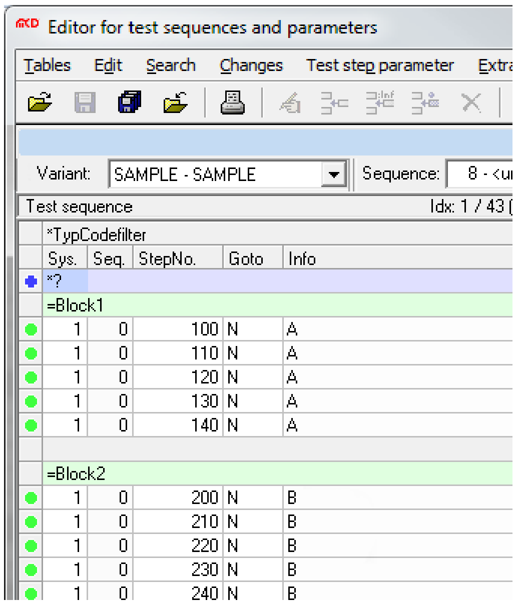
- Locking hardware used in a shared manner
- Optimization of the test process when test blocks will be performed using the hardware available
- Simultaneous, asynchronous testing of multiple devices under test
- Simultaneous testing of different devices under test (types)
- Synchronization of the test processes

To make this possible, the test will be divided into individual test blocks. For these test blocks, rules about how these test blocks can be processed and exchanged will be created.

Order number: # 150039

2. Configuring of Test Blocks

The test blocks are configured in Test Editor in TestManager. Test blocks can be created in Test Editor using a new filter.



Variant: SAMPLE - SAMPLE

*TypCodefilter

	Sys.	Seq.	StepNo.	Goto	Info
	*?				
	=Block1				
●	1	0	100	N	A
●	1	0	110	N	A
●	1	0	120	N	A
●	1	0	130	N	A
●	1	0	140	N	A
	=Block2				
●	1	0	200	N	B
●	1	0	210	N	B
●	1	0	220	N	B
●	1	0	230	N	B
●	1	0	240	N	B

Figure 1: Test blocks in their testing sequence

A test block and its block filter start with the equals sign “=” followed by any combination of text desired for the TestManager. This combination of text must comply with a pre - defined syntax, which will be described below, for working with ResourceManager.

A block filter will apply for subsequent test steps for the test sequence until a new or empty block filter has been defined. When working with ResourceManager, this approach may not be obvious and is not recommended for that reason.

3. Creating New Test Steps in TestManager

Three new system steps for the activation of the block filter and working ResourceManager have been introduced in TestManager:

- Block - Start
- Block - End
- Block - Change

3.1. Block - Start

The block - start step will be called after the device under test has started but before the actual test sequence begins.

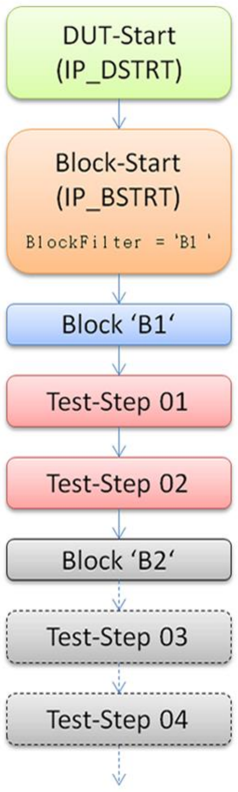


Figure 2: Block - start step

A block filter can be enabled in this block - start step. In the following test sequence, the step that matches the block filter will be started. Test steps that have not been assigned to a block filter will always be executed. If a block filter has not been set in the block - start step, all test steps will be executed.

In the flowchart above, the block filter is set to “B1” in the block - start step, which in turn will execute the “Test - Step 01” and “Test - Step 02” steps in the subsequent processing.

3.1.1. SetSeqBlock

A new command has been introduced in TestManager to set the block:

```
procedure SetSeqBlock(sBlockName : string);

sBlockName : Name of the active block filter
```

3.1.2. GetSeqBlock

The GetSeqBlock command serves to query the current block filter:

```
function GetSeqBlock : string;
```

3.1.3. ResetSeqBlock

The ResetSeqBlock command will reset the block filter:

```
procedure ResetSeqBlock;
```

3.1.4. GetSeqBlocks

The following command will query all defined blocks:

```
function GetSeqBlocks : stringvector;
```

This function will return a list of all blocks that have been defined.

3.1.5. Example for a Block - Start Step

In the simplest case, the block - start step will use the following structure:

```
step

    SetSeqBlock(‘B1‘);

end.
```

3.2. Block - End

Once the end of the test sequence has been reached, the block - end step will be called before the device under test process is terminated. Whether the testing process has been successfully completed or the test sequence should be run again can be determined in this block - end step.

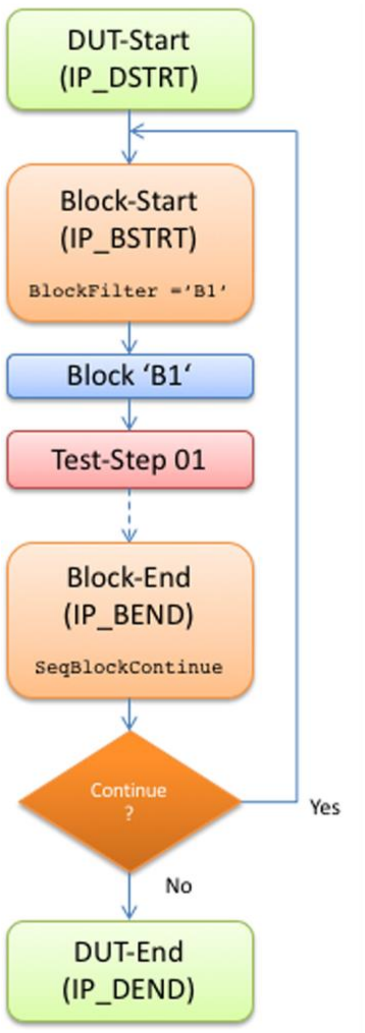


Figure 3: Block - end step

3.2.1. SeqBlockContinue

To determine if an additional block should be processed, a new command has been introduced in TestManager:

```
procedure SeqBlockContinue;
```

If this command is called, the block - start step and the test sequence will be processed again after the completion of the block - end step.

3.2.2. SeqBlockCanContinue

In exceptional cases, the test sequence will not be able to be executed at all. Such is the case, for example, when the device under test interrupts the test. In such cases, the “SeqBlockContinue” command may be executed so that this will not have any effect and the test will not be repeated.

The SeqBlockCanContinue command may be queried as follows:

```
function SeqBlockCanContinue : real;
```

The command will return “1” or “true”, if another block can be tested.

3.2.3. GetRemainingSeqBlocks

The following command can be used to query which blocks have not yet been processed:

```
function GetRemainingSeqBlocks : stringvector;
```

This function will return a list of all blocks that have yet to be executed.

3.2.4. Example for Block - End Step

In the simplest case, the block - end step will use the following structure:

```
Step
  if SeqBlockCanContinue and (Len(GetRemainingSeqBlocks) > 0) then begin
    SeqBlockContinue;
  end;
end.
```

3.3. Block - Change

Reacting when the current block changes is important in working with the ResourceManager. This will only occur in the block - start step during the normal test process. However, in step - by - step mode, the block may change at any time.

The block - change step was introduced in order to be able to respond to such changes. It will always be called when the current block changes.

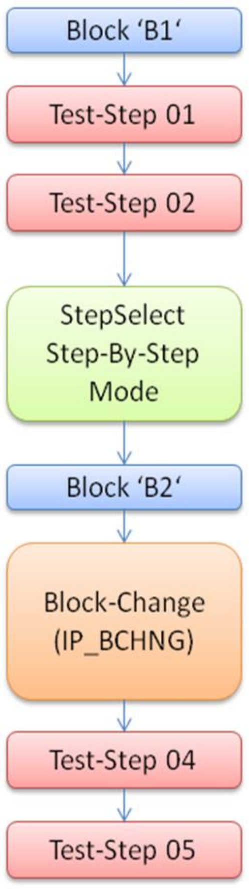


Figure 4: Block - change step

In the block - change step for example, ResourceManager can be notified about a forced change in the current block or processing will wait in the block - change step until ResourceManager has made the affected block available or released it, once all of the necessary resources are available.

4. Introduction to the Operation

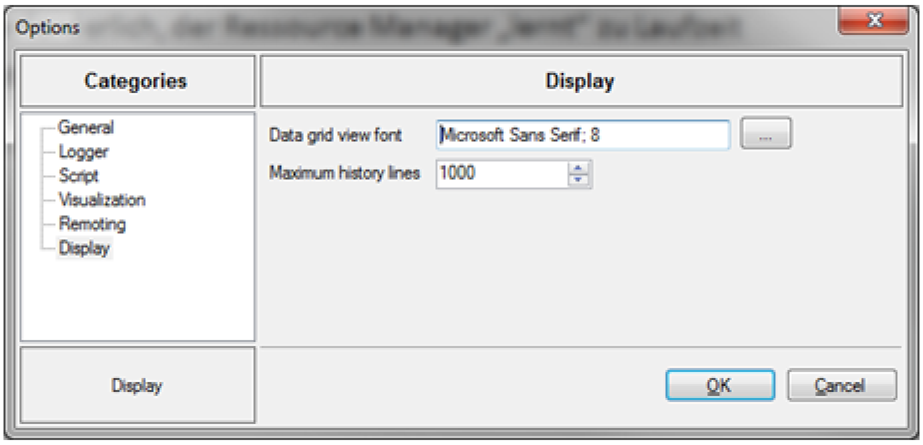
4.1. Quick Guide

As described in the foreword, ResourceManager serves to lock hardware mutually used and to release process blocks.

4.2. Setup

ResourceManager does not need to be configured. ResourceManager will “teach” itself about all of the instances, defined blocks and resources available at runtime.

Ultimately, the font used for display and the size of the history log can be adjusted to internal requirements as needed.



Data grid view font: Microsoft Sans Serif; 8

Maximum history lines: 1000

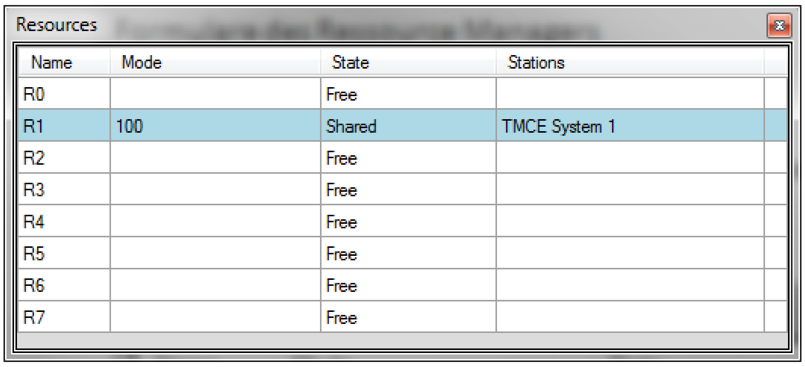
Figure 5: Setup of the ResourceManager

Like all Toolmonitors, the layout settings (form positions and column widths) will also be stored in the setup file.

4.3. ResourceManager Forms

4.3.1. Resources

All of the defined resources and their current state will be displayed on this window.



Name	Mode	State	Stations
R0		Free	
R1	100	Shared	TMCE System 1
R2		Free	
R3		Free	
R4		Free	
R5		Free	
R6		Free	
R7		Free	

Figure 6: Resources

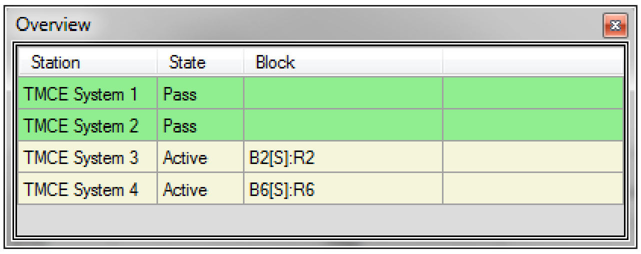
In ResourceManager, the resources can be assigned to an instance of TestManager in either exclusively or shared mode. If one resource will used in shared mode, it must be in a certain mode that can be shared, because otherwise resource handling will not be necessary.

The individual columns have the following meanings:

- Name (Name of the resource)
- Mode (Mode, when a resource will be shared simultaneously (shared state))
- State (Current status of the resource)
  It may be set to the following values:
  - Free
  - Shared
  - Exclusive
- Stations (Semi-colon separated list of the stations that are currently using the resource)

4.3.2. Overview

This window provides an overview of all stations and their current state.



Overview

Station	State	Block
TMCE System 1	Pass	
TMCE System 2	Pass	
TMCE System 3	Active	B2[S]:R2
TMCE System 4	Active	B6[S]:R6

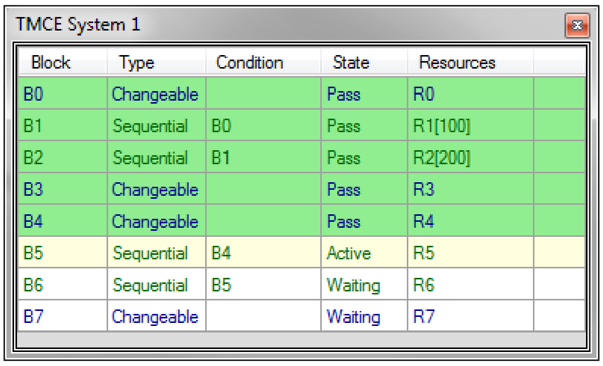
Figure 7: Overview

The individual columns have the following meanings:

- Station (Name of the TestManager instance)
- State (Current status)
  It may be set to the following values:
  - Start
  - Active
  - Waiting
  - Pass
  - Fail
  - Abort
- Block (Block being executed, if the station is active)

4.3.3. Stations

A form will be created for each instance of TestManager for the corresponding station.



TMCE System 1

Block	Type	Condition	State	Resources
B0	Changeable		Pass	R0
B1	Sequential	B0	Pass	R1[100]
B2	Sequential	B1	Pass	R2[200]
B3	Changeable		Pass	R3
B4	Changeable		Pass	R4
B5	Sequential	B4	Active	R5
B6	Sequential	B5	Waiting	R6
B7	Changeable		Waiting	R7

Figure 8: Stations

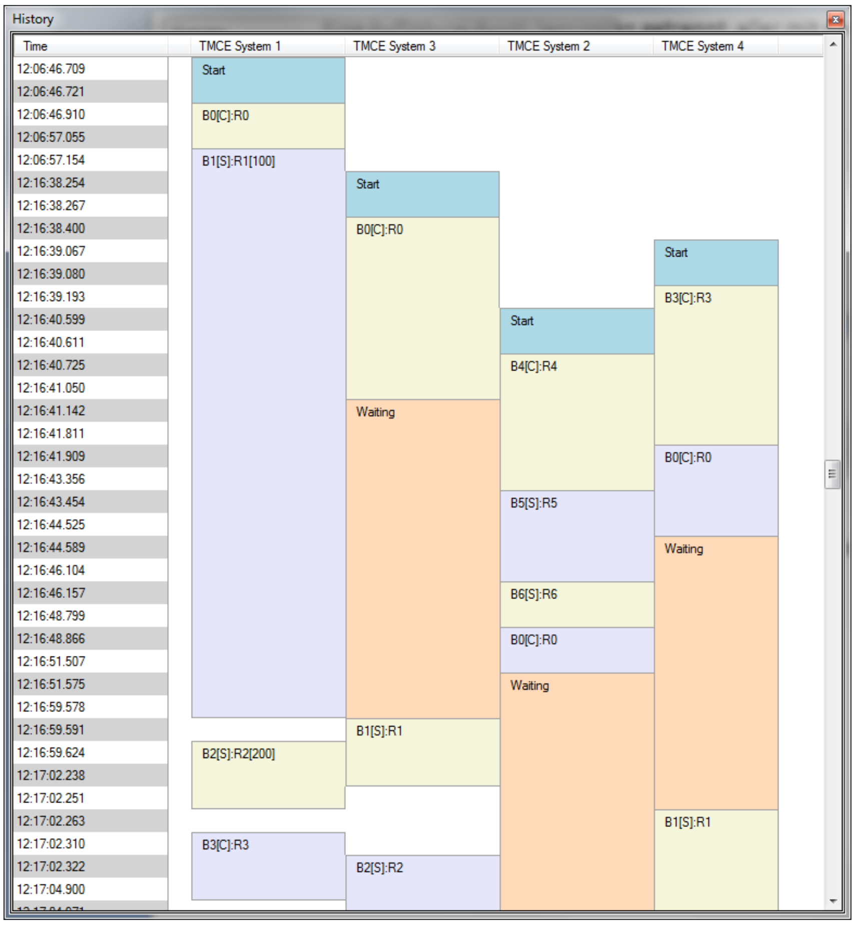
The test blocks for the current test sequences will be displayed in this window.

The individual columns have the following meanings:

- Block (Name of the blocks in the sequence list)
- Type (Type of block, a more precise description of the block types will be found in the documentation)
  It may be set to the following block types:
  - Fix
  - All
  - Changeable
  - Sequential
- Condition (Optional condition indicating when the block may be executed)
- State (Status of block)
  It may be set to the following values:
  - Waiting
  - Abort
  - Active
  - Pass
  - Fail
  - Omitted
- Resources (Semi-colon separated list with all of the resources associated with the block)

4.3.4. History

This window will display a chronological list of which block was processed when by which station or when a station had to wait for free resources.



History

Time	TMCE System 1	TMCE System 3	TMCE System 2	TMCE System 4
12:06:46.709	Start			
12:06:46.721				
12:06:46.910	B0[C]:R0			
12:06:57.055				
12:06:57.154	B1[S]:R1[100]			
12:16:38.254		Start		
12:16:38.267				
12:16:38.400		B0[C]:R0		
12:16:39.067				Start
12:16:39.080				
12:16:39.193				B3[C]:R3
12:16:40.599			Start	
12:16:40.611				
12:16:40.725			B4[C]:R4	
12:16:41.050				
12:16:41.142		Waiting		
12:16:41.811				
12:16:41.909				B0[C]:R0
12:16:43.356				
12:16:43.454			B5[S]:R5	
12:16:44.525				
12:16:44.589				Waiting
12:16:46.104				
12:16:46.157			B6[S]:R6	
12:16:48.799				
12:16:48.866			B0[C]:R0	
12:16:51.507				
12:16:51.575			Waiting	
12:16:59.578				
12:16:59.591		B1[S]:R1		
12:16:59.624	B2[S]:R2[200]			
12:17:02.238				
12:17:02.251				
12:17:02.263				B1[S]:R1
12:17:02.310	B3[C]:R3			
12:17:02.322		B2[S]:R2		
12:17:04.900				

Figure 9: History

5. Software Manual

5.1. Defining Process Blocks

The defined test blocks must comply with a pre-defined syntax when working with TestManager. The syntax uses the following structure:

BlockName[BlockType:Condition]:Resource[Mode]

In this structure, the “BlockType”, “Condition”, “Resource” and “Mode” settings are optional. The following sections will describe the previously defined block types.

5.1.1. Fixed Blocks (Fix | F)

One such process block will define a block that may first be executed once all previously defined blocks have been processed. In addition, blocks that follow after this block may not be processed before this block. Such a block therefore represents a fixed point in a process. If the “BlockType” has not been specified in the block definition, it will be treated as a fixed block.

- Syntax: F or Fix
- Following examples define fixed blocks that have not been bound to any resources:
    - Block01
    - Block01[F]
    - Block01[Fix]
- Following example defines a fixed block that has been bound to the “Res01” resource (Resource must be available in exclusive mode):
    - Block01[F]:Res01
- Following example defines a fixed block that has been bound to the “Res01” resource (Resource may be shared, however the mode must set to “100”. If the resource was free until this point, the mode will be set to “100”):
    - Block01[F]:Res01[100]

5.1.2. Fixed Blocks for all Stations (All | A)

One such process block will define a block that may first be executed once all previously defined blocks for all active stations (Station is not “PASS”, “FAIL” or “Disabled”) have been processed. In addition, blocks that follow after this block may not be processed before this block.

- Syntax: A or All
- Following examples define fixed blocks that have not been bound to any resources:
    - Block01[A]
    - Block01[All]
- Following example defines a fixed block that has been bound to the “Res01” resource (Resource must be available in exclusive mode):
    - Block01[A]:Res01
- Following example defines a fixed block that has been bound to the “Res01” resource (Resource may be shared, however the mode must set to “100”. If the resource was free until this point, the mode will be set to “100”):
    - Block01[A]:Res01[100]

5.1.3. Changeable Blocks (Changeable | C)

Such a block may be jumped over and executed at a later time, if the resources necessary for it are not available at that time. It is possible to define a conditional block. This conditional block however must be processed before the block may be executed.

- Syntax: C or Changeable
- Following examples define changeable blocks that have not been bound to any resources:
  - Block01[C]
  - Block01[Changeable]
- Following example defines a changeable block that has been bound to the “Res01” resource (Resource must be available in exclusive mode):
  - Block01[C]:Res01
- Following example defines a changeable block that has not been bound to any resources and that may first be executed after “Block01” has been completed:
  - Block02[C:Block01]
- Following example defines a changeable block that has been bound to the “Res01” resource and that may first be executed after “Block01” has been completed (resource (Resource must be available in exclusive mode):
  - Block02[C:Block01]:Res01
- Following example defines a changeable block that has been bound to the “Res01” resource (Resource may be shared, however the mode must set to “100”. If the resource was free until this point, the mode will be set to “100”):
  - Block01[C]:Res01[100]

5.1.4. Sequence - Blocks (Sequential | S)

Such a block must always be executed immediately after the block specified in the condition. Such a block can be jumped over if the block specified in the condition is jumped over.

- Syntax: S or Sequential
- Following examples define sequence blocks that have not been bound to any resources and must be executed immediately after the block defined before them:
  - Block01[S]
  - Block01[Sequential]
- Following example defines a sequence block that has been bound to the “Res01” resource and must be executed immediately after the block defined before this block (Resource must be available in exclusive mode):
  - Block01[S]:Res01
- Following example defines a sequence block that has not been bound to any resources and must be executed immediately after “Block01”:
  - Block04[S:Block01]

5.2. ResourceManager Commands

The following commands have been implemented in ResourceManager to synchronize process control with the TestManager.

5.2.1. SetBlockList

The test sequence configuration will be sent to ResourceManager using this command.

- Example (IP_DSTRT)

```
var
  sSystemString : string;
  vsBlocks : stringvector;

step

  vsBlocks := GetSeqBlocks;
  Debug.Show(1, vsBlocks);
  sSystemString := 'TMCE System ' + Str(globalvar.get('TMCE_System'));
  Toolmanager.SetValue('HWCONTROL', 'SetBlockList.' + sSystemString, vsBlocks);

end.
```

5.2.2. ResetBlockList

This command resets the configuration of the test sequence.

- Example (IP_RESET)

```
var
  sSystemString : string;

step

  sSystemString := 'TMCE System ' + Str(globalvar.get('TMCE_System'));
  Toolmanager.SetEvent('HWCONTROL', 'ResetBlockList.' + sSystemString);

end. .
```

5.2.3. DisableBlockList

This command disables a station (Instance of TestManager).

- Example (IP_RESET)

```
var
  sSystemString : string;

step

  sSystemString := 'TMCE System ' + Str(globalvar.get('TMCE_System'));
  if RegForm.Get('ENABLED') = '0' then begin
    Toolmanager.SetEvent('HWCONTROL', 'DisableBlockList.' + sSystemString);
  end else begin
    Toolmanager.SetEvent('HWCONTROL', 'ResetBlockList.' + sSystemString);
  end;

end.
```

5.2.4. GetNextBlock

The next block that should be processed will be queried using this command.

- Example (IP_BSTRT)

```
var
  sSystemString : string;
  sBlock : string;

step

  if Len(GetSeqBlocks) > 0 then begin

    sSystemString := 'TMCE System ' + Str(globalvar.get('TMCE_System', 0));
    while sbLock = '' do begin
      sBlock := Toolmanager.GetString('HWCONTROL', 'GetNextBlock.' + sSystemString);
      if sbLock = '' then begin
        DateTime.Delay(10, true);
      end;
    end;

    SetSeqBlock(sBlock);

  end;

end.
```

5.2.5. GetNextBlockAndPreferResources

The next block that should be processed will be queried using this command, whereby blocks that have been bound to resources will be given preference.

- Example (IP_BSTRT)

```
var
  sSystemString : string;
  sBlock : string;

step

  if Len(GetSeqBlocks) > 0 then begin

    sSystemString := 'TMCE System ' + Str(globalvar.get('TMCE_System', 0));
    while sbLock = '' do begin
      sBlock := Toolmanager.GetString('HWCONTROL', GetNextBlockAndPreferResources.' +
                sSystemString);
      if sbLock = '' then begin
        DateTime.Delay(10, true);
      end;
    end;

    SetSeqBlock(sBlock);

  end;

end.
```

5.2.6. SetBlockResult

At the end of a block, the test result will be set using this command.

- Example (IP_BEND)

```
var
  sSystemString : string;
  sBlock : string;

step

  if (Len(GetSeqBlocks) > 0) and (GetSeqBlock <> '') then begin

    sSystemString := 'TMCE System ' + Str(globalvar.get('TMCE_System', 0));

    if GetTestResult = 1 then begin
      Toolmanager.SetValue('HWCONTROL', 'SetBlockResult.' + sSystemString, 'Pass:' + GetSeqBlock);
    end else if GetTestResult = 3 then begin
      Toolmanager.SetValue('HWCONTROL', 'SetBlockResult.' + sSystemString, 'Break:' +
                          GetSeqBlock);
    end else begin
      Toolmanager.SetValue('HWCONTROL', 'SetBlockResult.' + sSystemString, 'Fail:' + GetSeqBlock);
    end;

    if SeqBlockCanContinue then begin

      Debug.Show(1, GetRemainingSeqBlocks);

      if (Len(GetRemainingSeqBlocks) > 0) or
         (Toolmanager.GetValue('HWCONTROL', 'AllBlocksFinished.' + sSystemString) = 0) then begin
        SeqBlockContinue;
      end;
    end;

  end;

end.
```

5.2.7. AllBlocksFinished

If all blocks have been processed can be queried using this command.

- Example (IP_BEND)

```
var
  sSystemString : string;
  sBlock : string;

step

  if (Len(GetSeqBlocks) > 0) and (GetSeqBlock <> '') then begin

    sSystemString := 'TMCE System ' + Str(globalvar.get('TMCE_System', 0));

    if GetTestResult = 1 then begin
      Toolmanager.SetValue('HWCONTROL', 'SetBlockResult.' + sSystemString, 'Pass:' + GetSeqBlock);
    end else if GetTestResult = 3 then begin
      Toolmanager.SetValue('HWCONTROL', 'SetBlockResult.' + sSystemString, 'Break:' + GetSeqBlock);
    end else begin
      Toolmanager.SetValue('HWCONTROL', 'SetBlockResult.' + sSystemString, 'Fail:' + GetSeqBlock);
    end;

    if SeqBlockCanContinue then begin

      Debug.Show(1, GetRemainingSeqBlocks);

      if (Len(GetRemainingSeqBlocks) > 0) or
         (Toolmanager.GetValue('HWCONTROL', 'AllBlocksFinished.' + sSystemString) = 0) then begin
        SeqBlockContinue;
      end;
    end;

  end;

end.
```

5.2.8. SetTestResult

The result of the entire test will be set using this command.

- Example (IP_DEND)

```
var
  sSystemString : string;

step

  sSystemString := 'TMCE System ' + Str(globalvar.get('TMCE_System', 0));

  if GetTestResult = 1 then begin
    Toolmanager.SetValue('HWCONTROL', 'SetTestResult.' + sSystemString, 'Pass');
  end else if GetTestResult = 3 then begin
    Toolmanager.SetValue('HWCONTROL', 'SetTestResult.' + sSystemString, 'Break');
  end else begin
    Toolmanager.SetValue('HWCONTROL', 'SetTestResult.' + sSystemString, 'Fail');
  end;

end.
```

5.2.9. GetBlock

A very specifically defined block can be requested using this command. This command will return “1” or “true” if the block can be executed.

- Example (IP_BCHNG)

```
var
  sSystemString : string;
  sBlock : string;


step

  if Len(GetSeqBlocks) > 0 then begin

    sSystemString := 'TMCE System ' + Str(globalvar.get('TMCE_System', 0));

    if IsSBS then begin
      if Toolmanager.GetValue('HWCONTROL', 'GetBlock.' + sSystemString + '.' + GetSeqBlock) = 0
then begin
        Screen.Dialog.Bitmap(':1');
        Screen.Dialog.Button('');
        Screen.Dialog.Button('');
        Screen.Dialog.SetText('Block change','Wait for resources for block ' + GetSeqBlock);
        Screen.Dialog.Show(false, 1);
      end;
    end;

    while Toolmanager.GetValue('HWCONTROL', 'GetBlock.' + sSystemString + '.' + GetSeqBlock) = 0
do begin
      DateTime.Delay(10, true);
    end;

    Screen.Dialog.Hide;

  end;

end.
```

5.2.10. SetBlock

A very specific block can be set using this command, even if the assigned resource has not been released. The status of the resource will be changed from “Exclusive” to “Shared”, if necessary.

- Example (IP_BCHNG)

```
var
  sSystemString : string;
  sBlock : string;


step

  if Len(GetSeqBlocks) > 0 then begin

    sSystemString := 'TMCE System ' + Str(globalvar.get('TMCE_System', 0));

    Toolmanager.SetValue('HWCONTROL', 'SetBlock.' + sSystemString, GetSeqBlock);

  end;

end.
```