# Table of Contents

1 winIDEA Workspace Configuration
   1.1 Master winIDEA Workspace Configuration
   1.2 Slave winIDEA Workspace Configuration

2 winIDEA Analyzer Configuration for Synchronized Trace
   2.1 Recorder (i.e. iCS700) Configuration
   2.2 AURIX-specific Manual Trace Configuration (i.e. MCDS Configuration)

3 Synchronized Debug Operation
   3.1 Initial State after Download
   3.2 Run / Stop / Breakpoint Operation
   3.3 Single-Step Operation

4 Synchronized Trace Operation

5 Profiler Timeline View

6 Technical support
   6.1 Contact Region Support
   6.2 Email the International Support Team
   6.3 Call the International Support Team
1 winIDEA Workspace Configuration

For synchronous debug and trace of two AURIX devices, two winIDEA workspaces need to be created, a “master” workspace and a “slave” workspace. Which one of the two AURIX devices acts as a master or as slave, is, in terms of synchronous debug/trace operation, an arbitrary selection by the user.

In the following example, we assume that the “master” device uses an iC5700 + Active Probe Infineon AGBT. The “slave” device uses an iC5700 + Active Probe Infineon DAP.

In addition, each iC5700 is equipped with an FNET Hub and a CAN2/LIN2 Add-On module. Both iC5700 FNET Hubs are connected via the FNET SYNC cable. Figure 1 depicts the overall setup.
1.1 Master winIDEA Workspace Configuration

The overall winIDEA configuration starts with the Master winIDEA Instance. Launch winIDEA and create a basic AURIX Debug and Trace workspace.

The following workspace configurations are relevant for sync debug and trace.

1.1.1 Infineon AGBT Active Probe Detection

After the communication to the iC570 has been established, it is recommended to perform a detection of the connected Active Probe. This can be done via the menu “Hardware – Emulation Options – Probe”. Select Active Probe and then click the “Refresh” button. Select the detected AGBT Active Probe.

![Emulation Options](Image)

Figure 2: AGBT Active Probe Detection

1.1.2 DAP Clock Frequency

The DAP clock frequency can be set in the “CPU Setup - SoC” dialog. This dialog can be opened via the menu “Hardware – CPU Options...”.

The DAP clock frequency should be set to a high clock rate for an optimized target status polling rate. A high target polling rate improves the sync debug latencies as well as the time synchronization between on-chip timestamp and BlueBox timestamp.

In Figure 3 the DAP clock is set to 100MHz. The maximum possible clock rate mainly depends on the DAP signal routing on the target board.

![CPU Setup](Image)

Figure 3: DAP Clock Configuration
1.1.3 FNET Configuration

For synchronized debug/trace, at least an Active Probe (either DAP or AGBT) needs to be connected to the iC5700 via FNET. The Active Probes should be connected to the FNET connector on the front-side of the iC5700 Base Unit. Optionally, other Add-On modules can be connected to the FNET connectors of the iC5700 Hub.

The complete FNET configuration of an iC5700 can be set in the “Options – FNet” dialog, which can be opened via the “Hardware – Options…” menu.

A good start for creating a new configuration is to push the “Refresh” button on the “Currently connected FNodes” section. This updates the list of currently connected FNodes, i.e. Active Probes and AOMs.

From this list either individual FNodes can be included the “FNode configuration” via the “Add >” button or all connected FNodes can be added by clicking the “Create configuration for connected FNodes” button.

![FNET Configuration](image)

Figure 4: Sample FNET Configuration (Master winIDEA Instance using AGBT Active Probe and CAN/LIN AOM)

1.1.4 BlueBox Synchronization

Next step is to establish the master-slave connection between the two winIDEA workspaces, the master and the slave workspace. This is done in the “Options – BlueBox Sync” dialog, opened via the “Hardware – Options…” menu.

Figure 5 shows a sample BlueBox Sync configuration of a Master winIDEA instance. This winIDEA instance is selected to the Master by choosing “This winIDEA instance is synchronization master”. The winIDEA workspace of the Slave winIDEA instance can be selected via “Slave workspaces - Add...”. 
Synchronous Debug & Trace on two Infineon AURIX Devices

Pushing the “Preset Probe and SoC Configuration” causes winIDEA to automatically adjust the Active Probe and SoC configurations needed for synchronized debug and trace.

The “Synchronize time” tick box must be enabled for synchronized trace.

![Sample Master winIDEA BlueBox Sync Dialog](image)

The above settings become active in the iC5700, the Active Probe, AOMs and on the target, by performing a Download (menu: »Debug – Download«) or Symbol Download (menu: »Debug – Load Symbols only«).

The download operation also launches the Slave winIDEA Instance.

After successful download and launch, the Master winIDEA instance should be in state »STOP*«, the Slave winIDEA instance in state »ONLINE (ready to ATTACH)«.

The ‘*’ in the winIDEA status indicates the synchronized debug is not now active.

Figure 6 depicts a Master and Slave winIDEA instance after a successful download operation in the Master winIDEA instance.
1.1.5 FNET Operation

Finally, the “FNET Operation” setting for the Active Probes should be checked. Open the “FNET Operation” dialog via the menu “Hardware – FET Operation…”. As shown in Figure 7, the “Qualifier…” should be set to “Start:Enabled”. In addition, all four tick boxes for “Stop on StopSync”, “Run on RunSync”, “Use fast status polling” and “Generate StopSync when stopped” must be enabled to allow for synchronized debug.

Figure 7: FNET Operation Dialog for an AGBT Active Probe supporting Synchronized Debug


1.2 Slave winIDEA Workspace Configuration

1.2.1 DAP Active Probe Detection

After the communication to the ic570 has been established, it is recommended to perform a detection of the connected Active Probe. This can be done via the menu “Hardware – Emulation Options – Probe”. Select Active Probe and then click the “Refresh” button. Select the detected DAP Active Probe. In the Active Probe detection shown in Figure 8, the Active Probe has been given a alias “DAP_SLV”.

![Emulation Options](image)

Figure 8: ic5700 Active Probe Detection.

1.2.2 DAP Clock Frequency

The DAP clock frequency can be set in the “CPU Setup - SoC” dialog. This dialog can be opened via the menu “Hardware – CPU Options…”

The DAP clock frequency should be set to a high clock rate for an optimized the target status polling rate. A high target polling rate improves the sync debug latencies as well as the time synchronization between on-chip timestamp and BlueBox timestamp.

In Figure 9 the DAP clock is set to 100MHz. The maximum possible clock rate mainly depends on the DAP signal routing on the target board.

![CPU Setup](image)

Figure 9: DAP Clock Configuration

1.2.3 FNET Configuration

For synchronized debug/trace, at least an Active Probe (either DAP or AGBT) needs to be connected to the ic5700 via FNET. The Active Probes should be connected to the FNET connector on the front-side of the ic5700 Base Unit. Optionally, other Add-On modules can be connected to the FNET connectors of the ic5700 Hub.
The complete FNET configuration of an iC5700 can be set in the “Options – FNet” dialog, which can be opened via the “Hardware – Options…” menu.

A good start for creating a new configuration is to push the “Refresh” button on the “Currently connected FNodes” section. This updates the list of currently connected FNodes, i.e., Active Probes and AOMs.

From this list either individual FNodes can be included the “FNode configuration” via the “Add >” button or all connected FNodes can be added by clicking the “Create configuration for connected FNodes” button.

![Options Dialog]

Figure 10: Sample FNET Configuration (Slave winIDEA Instance using DAP Active Probe and CAN/LIN AOM)

### 1.2.4 BlueBox Synchronization

Next step is to establish the master-slave connection between the two winIDEA workspaces, the master and the slave workspace. This is done in the “Options – BlueBox Sync” dialog, opened via the “Hardware – Options…” menu.

Figure 11 shows a sample BlueBox Sync configuration of a Slave winIDEA instance. This winIDEA instance is selected to the Slave by choosing “Allow slave-sync operation”.

Pushing the “Preset Probe and SoC Configuration” causes winIDEA to automatically adjust the Active Probe and SoC configurations needed for synchronized debug and trace.

The “Synchronize time” tick box must be enabled for synchronized trace.
The above settings become active in the iC5700, the Active Probe, AOMs and on the target, by performing a Download (menu: »Debug – Download«) or Symbol Download (menu: »Debug – Load Symbols only«).

After successful download and launch, the Slave winIDEA instance should be in state »**STOP*« (same as Master winIDEA instance).

The ‘*’ in the winIDEA status indicates the synchronized debug is not now active.

Figure 12 depicts a Master and Slave winIDEA instance after a successful download operation in the Slave winIDEA instance.
1.2.5  FNET Operation

Finally, the “FNET Operation” setting for the Active Probes should be checked. Open the “FNET Operation” dialog via the menu “Hardware – FET Operation...”.

As shown in Figure 7, the “Qualifier…” should be set to “Start:Enabled”. In addition, all four tick boxes for “Stop on StopSync”, “Run on RunSync”, “Use fast status polling” and “Generate StopSync when stopped” must be enabled to allow for synchronized debug.

Figure 13: FNET Operation Dialog for a DAP Active Probe supporting Synchronized Debug
2  winIDEA Analyzer Configuration for Synchronized Trace

2.1  Recorder (i.e. iC5700) Configuration

As depicted in Figure 14 and Figure 15, the Analyzer should enable “Profiler” and “Manual Trigger/Recorder configuration”.

In addition, the iC5700 trace recorder must be set to:

- Start: On Trigger
- Timer Interpolation: enabled
- Generate time synchronization messages: enabled
- (for DAP) Upload while sampling: enabled

![Analyzer Configuration - [DT_OS_Tasks_Run_UWS_TSUREL_CAN]](image)

Figure 14: iC5700 Recorder Configuration for Synchronized Trace using AGBT Active Probe

![Analyzer Configuration - [DT_OS_Tasks_Run_UWS_TSUREL_CAN]](image)

Figure 15: iC5700 Recorder Configuration for Synchronized Trace using DAP Active Probe
2.2 AURIX-specific Manual Trace Configuration (i.e. MCDS Configuration)

**Trigger Position:**
As the Emulation Memory of the AURIX is used as FIFO for the trace data streaming (Upload-While-Sampling) to the iC5700, the Trigger Position can be set to »End«.

**MUX:**
The POB and BOB Muxes must be set as appropriate for the individual trace use-case.

**Time stamps:**
The time stamp source must be »tsu_rel«.
The TSU_REL Prescaler value (TSUPRSCl) may be chosen as appropriate for the individual trace use-case. A TSUPRSCl value of 1 yields the maximum trace timing accuracy.
**NOTE:** Synchronized trace only works when using TSU_REL time stamping. TICK time stamping is not supported for synchronized trace.

Figure 16 shows a sample MCDS configuration supporting synchronized trace.

![MCDS Configuration Table](image)
MCDS Multi-Core Cross Connect (MCX) Configuration

The MCX module of MCDS must be configured for TSU_REL time stamping.

In the sample MCX configuration shown in Figure 17, the MCX counter 15 in conjunction with EVT23 is used to enable TSU_REL time stamp messaging only while the core (attached to POB X) is executing instructions (i.e., is running). EVT8 is connected to the tsu_tc_trig (i.e., to the TSUPRSCL prescaler). This means, a TSU_REL time stamp sync message is generated periodically upon every underflow of the TSUPRSCL prescaler.

In addition, the MCX action »trace_done« is set to NEVER, which is required for »Upload-While-Sampling« mode.

Figure 17: Sample MCX Configuration using TSU_REL time stamping required for Synchronized Trace
3 Synchronized Debug Operation

This section describes the basic synchronized debug operations.

3.1 Initial State after Download

After a download operation on both Master and Slave winIDEA instance, both devices are in »STOP« state (see Figure 18). The '*' indicates that synchronized debug is currently active.

3.2 Run / Stop / Breakpoint Operation

Starting the CPU (»Debug – Run Control – Run (F5)«) on the Master, also causes the Slave to enter »RUN« state (see Figure 19).
A Breakpoint hit on the Master also causes the Slave to halt. The CPU enters the AURIX-specific »SUSPENDED« state (see Figure 20).

Starting the CPU (RUN) on the Slave, also causes the Master to enter »RUN« state (see Figure 21).
A Breakpoint hit on the Slave also causes the Master to halt. The CPU enters the AURIX-specific »SUSPENDED« state (see Figure 22).

Figure 22: A Breakpoint hit on the Slave, causes the Master to enter »SUSPENDED« State.

3.3 Single-Step Operation

Single-Step operations on the Master do not have any effect on the Slave. The Slave remains in »SUSPENDED« state. While in »SUSPENDED« state, single-step operations (source-level or assembly-level) are not possible. Single-Step is only possible when the SoC is in »STOP« state.
4 Synchronized Trace Operation

Performing a synchronized trace recording requires the following steps:

1. **Alignment of iC5700 BlueBox Time Offset.** This needed for the trace visualization in the Profiler Timeline View. The Time Offset alignment aligns the time offset from time 0 to the first trace recording, in each Profiler Timeline. It has not influence on the overall trace synchronization accuracy.

2. **Start of both Trace Recorders.** The operations starts the generation of trace messages on the target. However, as the recorder (iC5700) is set to »ON Trigger« the iC5700 will not record any trace data until the occurrence of the Trace Trigger.

3. **Trace Trigger.** A trace trigger can be injected either on the Slave or on the Master winDEA instance. The trigger is propagated via the FNET to both Master and Slave iC5700 trace recorder. The propagation delay of the trace trigger via FNET has no influence on the overall trace synchronization accuracy.

The sample Python script shown in performs the steps described above.

```python
import os
import sys
import isystem.connect as ic

mydir = <User-specific Workspace Location>

# Master Workspace location
workspaceLocation1 = os.path.join(mydir, 'Demo1/winidea/Tasks_ISR2_States_NoOnInstr', 'demo1_mst_agbt.xjrf')
trdName1 = 'demo1_mst_agbt.trd'

# Slave Workspace location
workspaceLocation2 = os.path.join(mydir, 'Demo2/winidea', 'demo2_slv.xjrf')
trdName2 = 'demo2_slv.trd'

# Create all necessary Objects to control the Master.
cmgr_MST = ic.ConnectionMgr()
connectionConfig = ic.CConnectionConfig()
connectionConfig.workspace(workspaceLocation1)
cmgr_MST.connect(connectionConfig)
debug_MST = ic.CDebugFacade(cmgr_MST)
ideCtrl_MST = ic.CIDEController(cmgr_MST)
traceDoc_MST = ic.CTraceController(cmgr_MST, trdName1, 'w')

# Create all necessary Objects to control the Slave.
cmgr_SLV = ic.ConnectionMgr()
connectionConfig = ic.CConnectionConfig()
connectionConfig.workspace(workspaceLocation2)
cmgr_SLV.connect(connectionConfig)
debug_SLV = ic.CDebugFacade(cmgr_SLV)
ideCtrl_SLV = ic.CIDEController(cmgr_SLV)
traceDoc_SLV = ic.CTraceController(cmgr_SLV, trdName2, 'w')

# Adjust BlueBox (iC5700) Time Offset (just needed for aligned display in Profiler Timeline).
```
outParams = ic.StrStrMap()
inParams  = ic.StrStrMap()
ideCtrl_MST.serviceCall("/IOPEN/HW.HW.GetBBTime", inParams, outParams)
timeValue = outParams['Time']

# Adjust Time Offset of Master BlueBox.
ideCtrl_MST.setOption_str('/Document/' + trdName1 + '/Loader.HW.Data.TimerOffsetActivation', timeValue)

# Adjust Time Offset of Master BlueBox.
ideCtrl_SLV.setOption_str('/Document/' + trdName2 + '/Loader.HW.Data.TimerOffsetActivation', timeValue)

# Start Trace Recorder on Master and Slave. Both wait for a common Trace Trigger.
traceDoc_MST.start()
traceDoc_SLV.start()

# Start Master CPU. This also start Slave CPU due to sync debug setup.
debug_MST.run()

# Inject SYNC Trace start on Trigger Channel 1 (i.e. TraceTrig).
ideCtrl_MST.serviceCall("/IOPEN/HW.FNet.FTrig_Inject", "TC:1")

# Done. Disconnect from both Workspaces again.
cmgr_SLV.disconnect()
cmgr_MST.disconnect()

Listing 1: Sample Python Script to start a synchronized Trace Operation
As stated above, the trace trigger is propagated throughout the entire system via FNET. In addition to the trace trigger, also a trace clock is propagated to all trace recorders. Synchronized debug start/stop operations are as well controlled via FNET.

A simplified Block Diagram of the Overall iSYSTEM FNET Concept is depicted in Figure 23.
5 Profiler Timeline View

Figure 24 shows a sample Profiler Timeline view of a synchronized dual-AURIX trace recording. Besides the trace of the AUTOSAR OS tasks and ISR2s, also the CAN frame are recorded on both Socs (also synchronized to AURIX MCDS trace).

Figure 24: Sample Profiler Timeline View of a Synchronized Trace on two AURIX Devices (top = Master, bottom = Slave)
Figure 25 zooms into a specific section of the Profiler Timeline view of Figure 24. The Profiler Timeline on the top shows the OS ISR2 and CAN bus activities on the Master. The Profiler Timeline on the bottom shows the OS thread (task or ISR2) and CAN bus activities on the Slave. The scenario shown here is the following:

1. The Master sends out a CAN frame with ID 0x110. Transmit completion causes a TX ISR2, CANSR0_ISR (blue cursor).

2. The Slave receives the CAN frame 0x110, issues an RX interrupt, CANSR4_ISR, which spawns an OS thread (i.e., task), Rte_Event_Task. A Runnable within this task (not shown here) responds by sending out a CAN frame with ID 0x111. Transmit completion causes a TX ISR2, CANSR0_ISR (yellow cursor).

3. The Master receives this CAN frame 0x111 and issues an RX interrupt, CANSR4_ISR (yellow cursor).
6 Technical support

6.1 Contact Region Support

Germany (AT-CH-DE)
iSYSTEM AG
Phone: +49(8138)697130
Email: support@isystem.com

Slovenia
iSYSTEM Labs d.o.o.
Phone: +386(1)5680666
Email: support@isystem.si

United States of America
iSYSTEM USA
Phone: +1(858)3859100
Email: supportusa@isystem.com

China
Flash Technology Trading (Shanghai) Limited
Phone: +862161457130
Email: sales@flashtech.com.cn

Singapore, Malaysia, Philippine, Thailand
Flash Technology Pte Limited
Phone: +6567496168
Email: sales@flashtech.com.sg

India
Trident Infosol Pvt. Ltd.
Phone: +91(80)42878787
Email: esg.fae@tridentinfosol.com

6.2 Email the International Support Team

support@isystem.com

Our technical support team will respond to your email within 24 hours.

6.3 Call the International Support Team

International +38615680666
Germany +49 8138 6971 30

Please note that the above telephone numbers are only available weekdays between 8 AM and 3 PM UTC+1.

iSYSTEM has made every effort to ensure the accuracy and reliability of the information provided in this document at the time of publishing. Whilst iSYSTEM reserves the right to make changes to its products and/or the specifications detailed herein, it does not make any representations or commitments to update this document.

© iSYSTEM. All rights reserved.