

Virtual Analysis of an i.MX6 Multicore SoC Design

Jon McDonald
Solutions Architect
Mentor Consulting Division

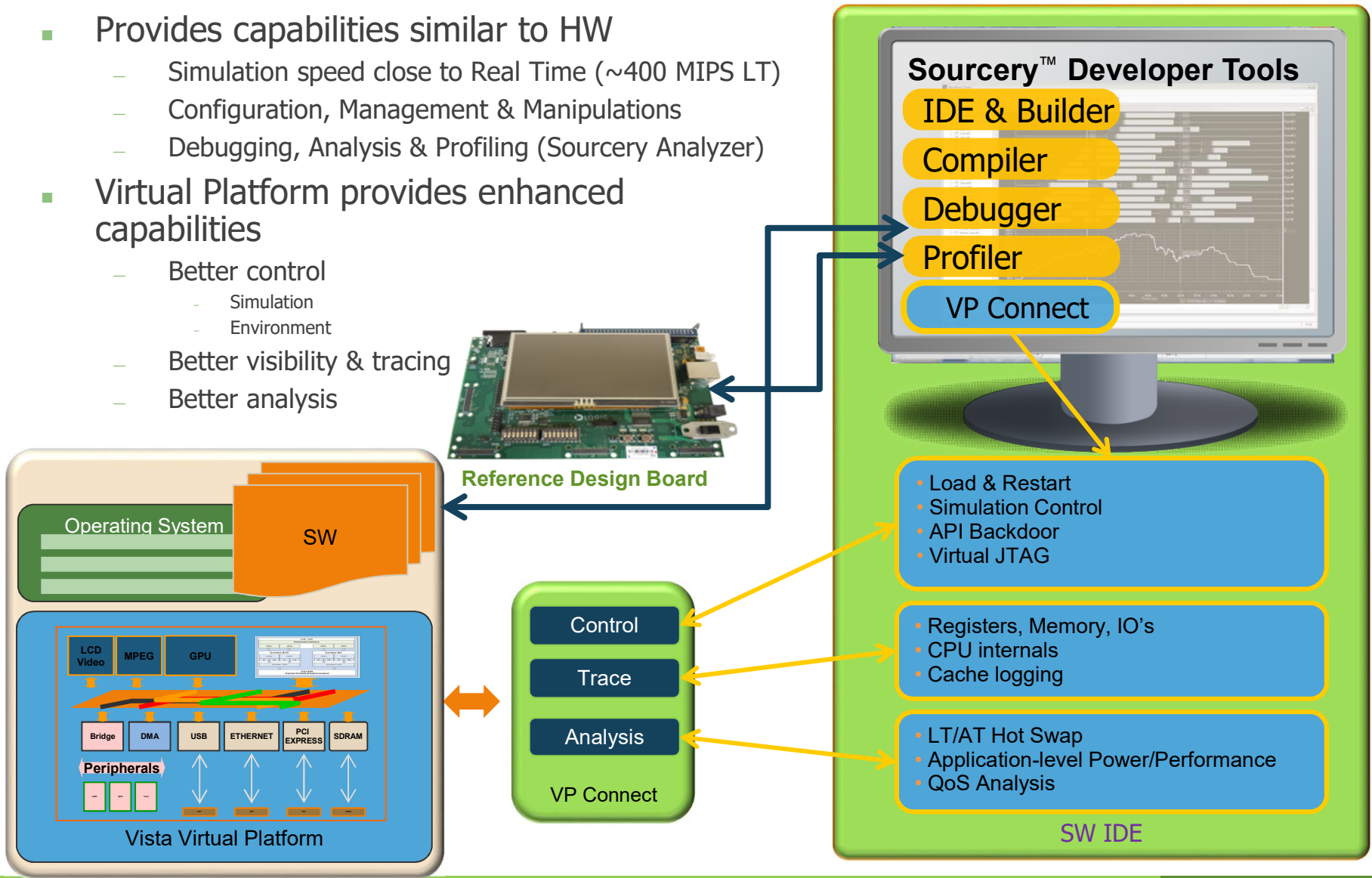


Android is a trademark of Google Inc. Use of this trademark is subject to Google Permissions.
Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Qt is a registered trade mark of Digia Plc and/or its subsidiaries. All other trademarks mentioned in this document are trademarks of their respective owners.

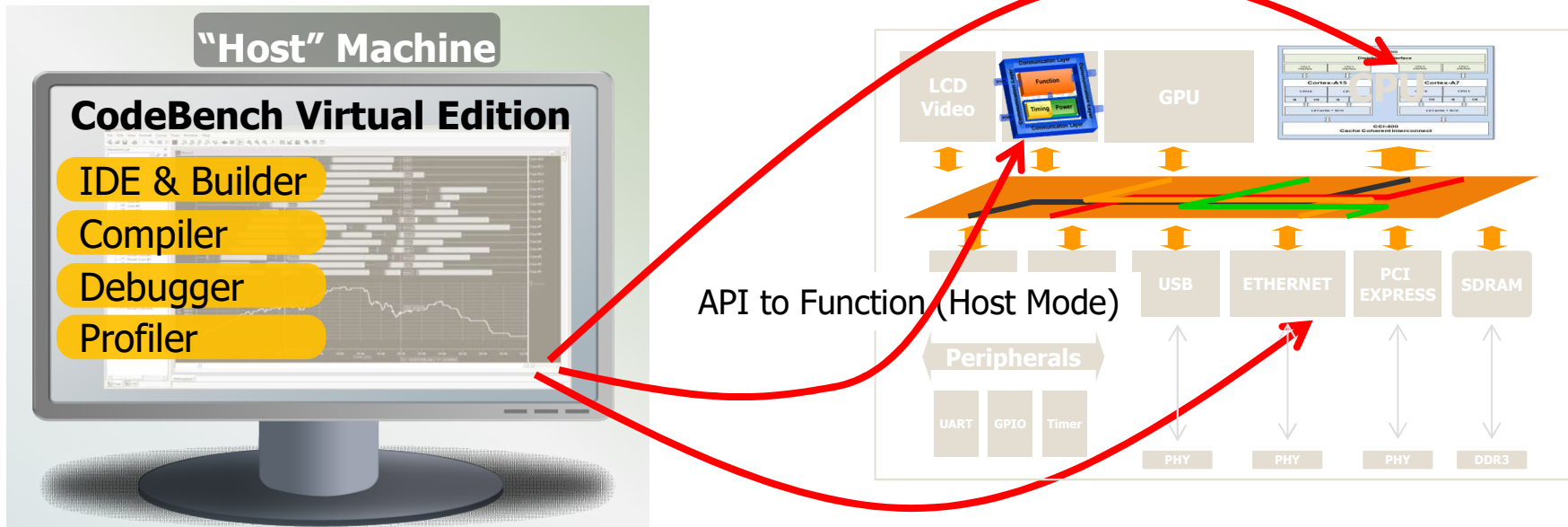
Virtual Platform Approach

- Provides capabilities similar to HW
 - Simulation speed close to Real Time (~400 MIPS LT)
 - Configuration, Management & Manipulations
 - Debugging, Analysis & Profiling (Sourcery Analyzer)
- Virtual Platform provides enhanced capabilities
 - Better control
 - Simulation
 - Environment
 - Better visibility & tracing
 - Better analysis



Connection to Virtual Target

GDB Connection to CPU (Bare-metal debug)

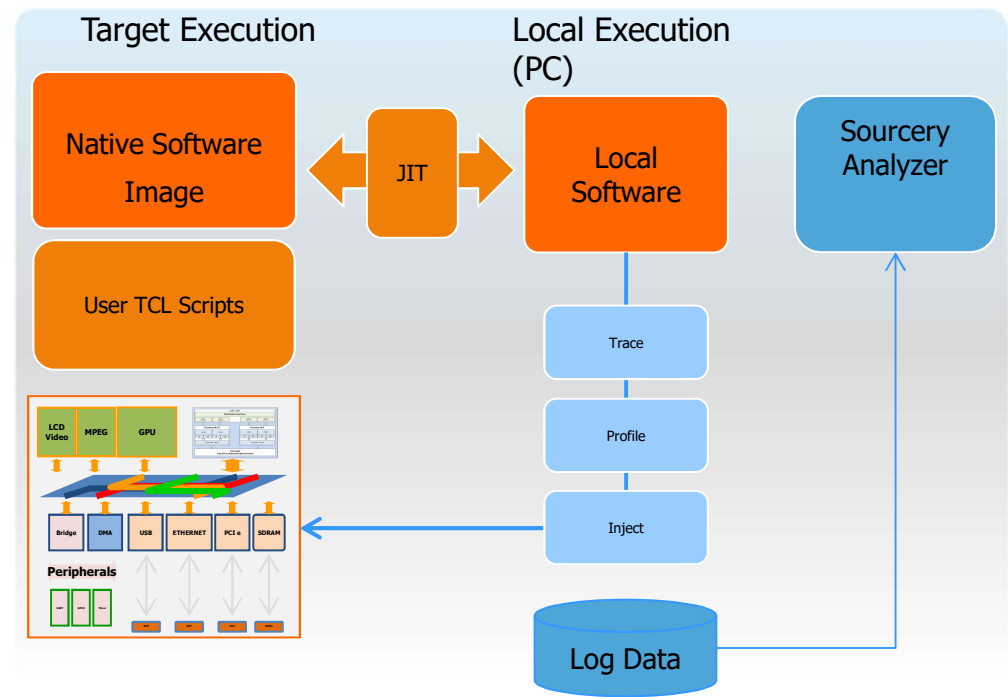


Ethernet Connection to OS (Linux Mode)

The debugger can connect to the virtual target in Linux, Bare-metal or Host mode without any probe.

Non-intrusive technology

- Debug and Profile unmodified software images
 - Code Instrumentation affects behavior and performance
- Debug third party Software when sources are not available
- Manipulate software dynamically at runtime



Non-intrusive Analysis

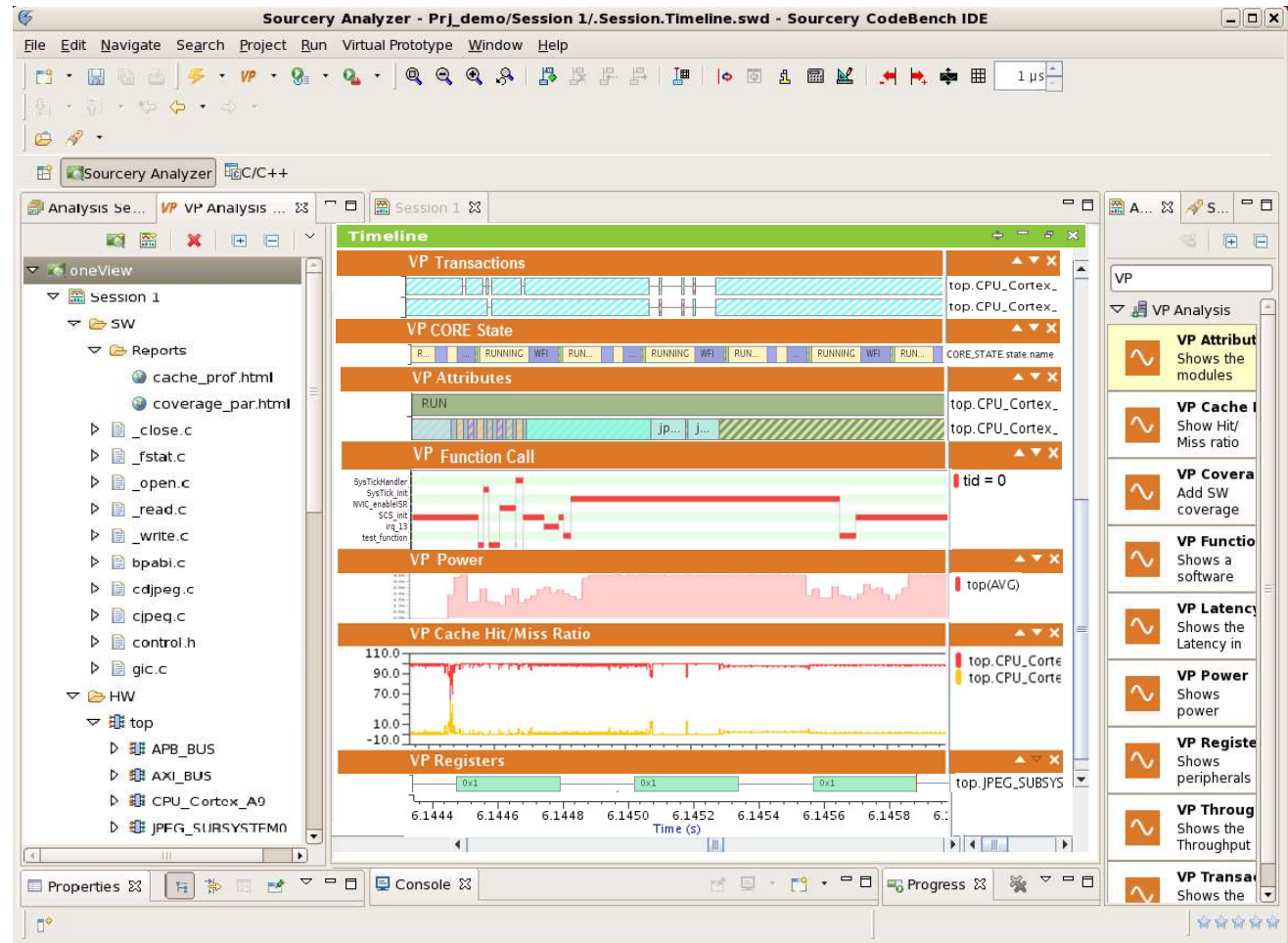
- Concurrent HW & SW Analysis under CodeBench VE – Sourcery Analyzer

Hardware:

- ✓ Transactions
- ✓ Throughput,
- ✓ Latency
- ✓ Power
- ✓ Registers
- ✓ Cache Hit/Miss

Software:

- ✓ Function-calls
- ✓ Profiling
- ✓ Code-Coverage
- ✓ CPU States
- ✓ CPU Current-function
- ✓ OS-Aware:
 - Tasks
 - Interrupts
 - Trap
 - ...
- ✓ Custom HW/SW Analysis



What Are We Designing?

- A yacht water depth alarm system with GPS information panel
- Using Freescale i.MX6 technology with I2C peripherals for Sonar, GPS, and the Alarm
- The alarm is raised when the water depth becomes too low
- The water depth and GPS details along with speed are displayed on the LCD panel



Hardware Components

- Freescale i.MX6
 - LCD
- I2C Devices
 - GPS
 - Sonar
 - Alarm



Simulation Infrastructure

- Virtual i.MX6, including:

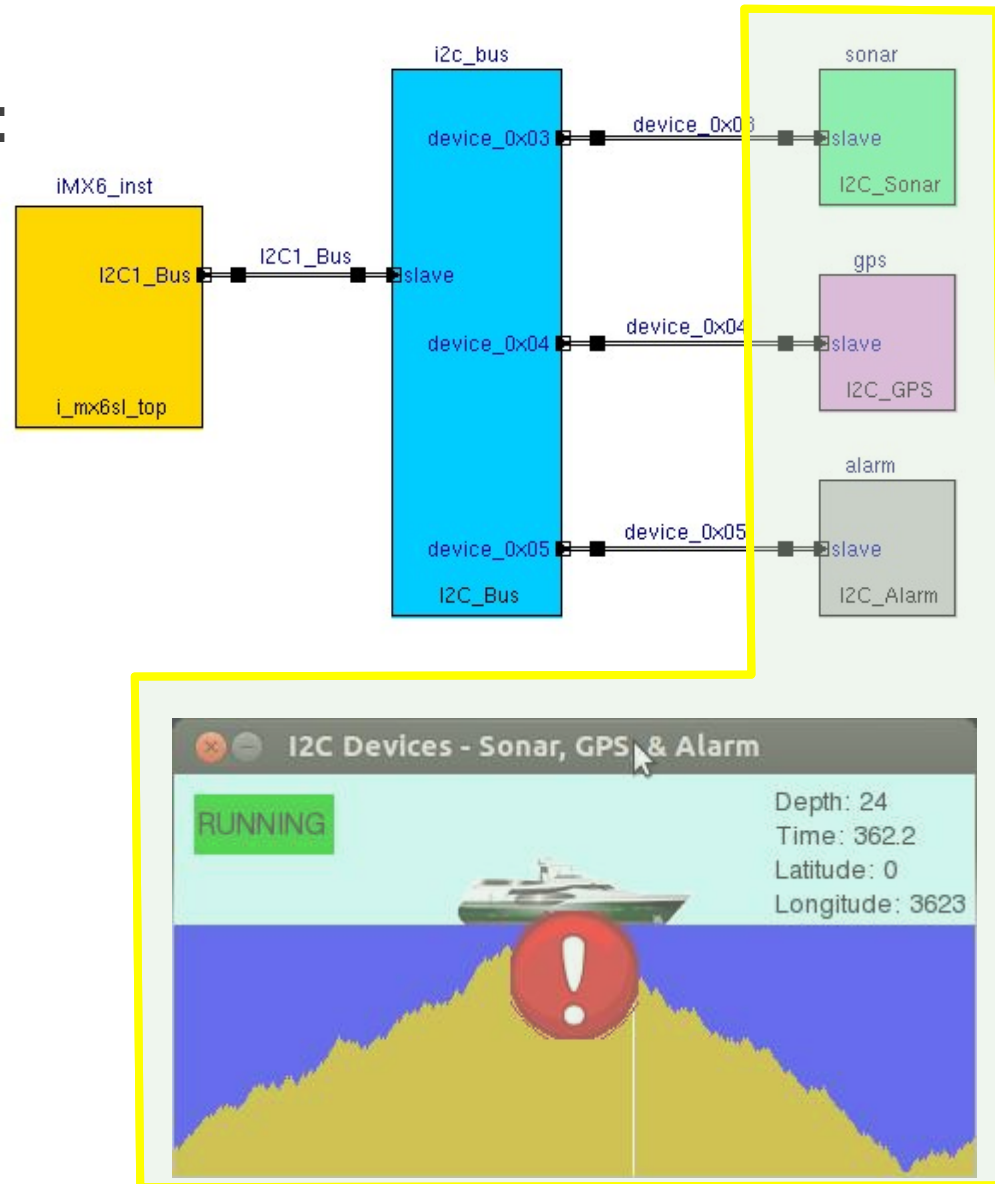
- LCD
- I2C, UARTs, SDCard

- Virtual I2C Devices

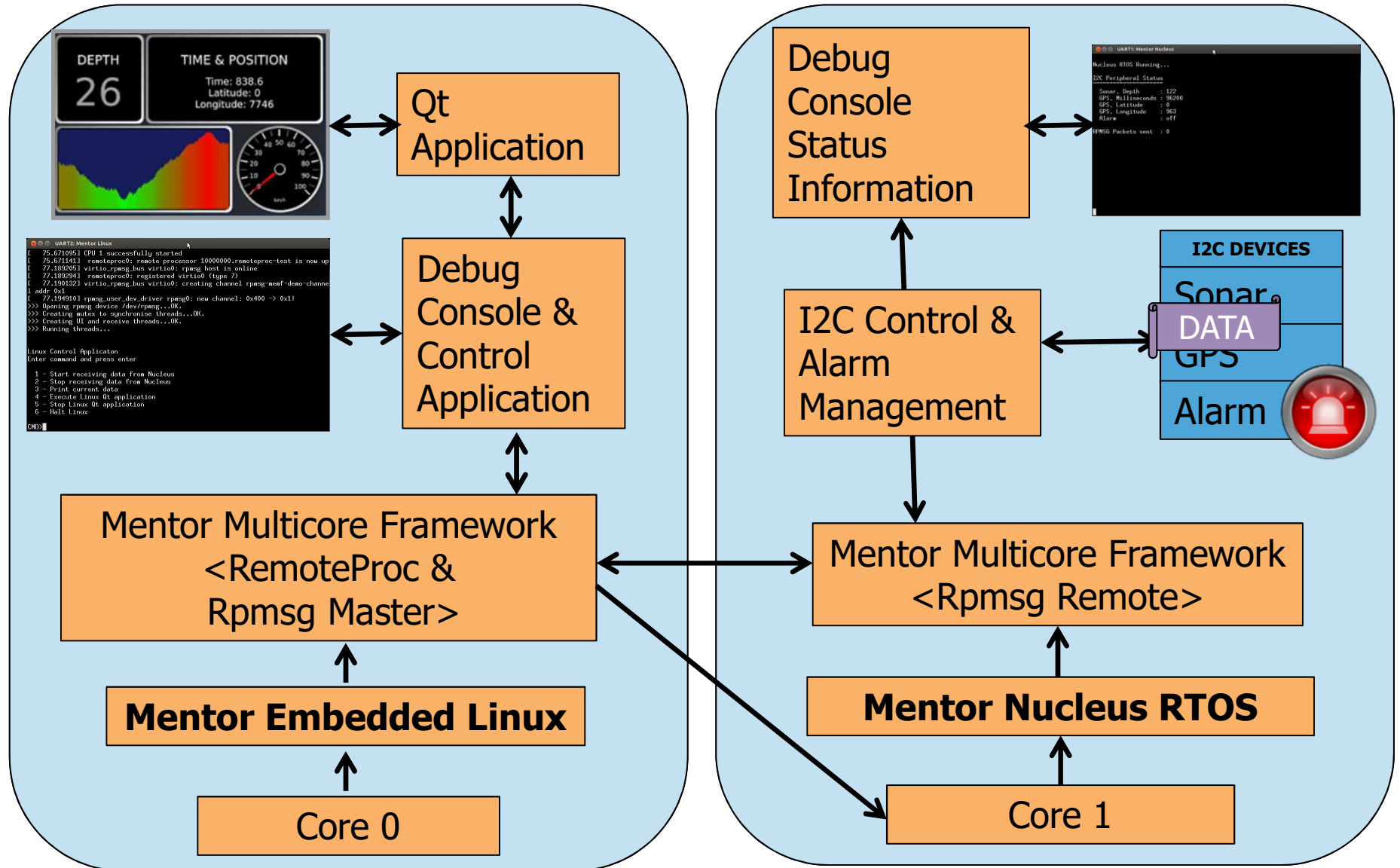
- GPS
- Sonar
- Alarm

- Simulation Stimulus

- Stop/Start yacht
- Generate random water depth
- Generate GPS data
- Display Alarm state



Embedded Software Components



Running the Virtual Platform

The screenshot displays the XTerm environment with three main windows:

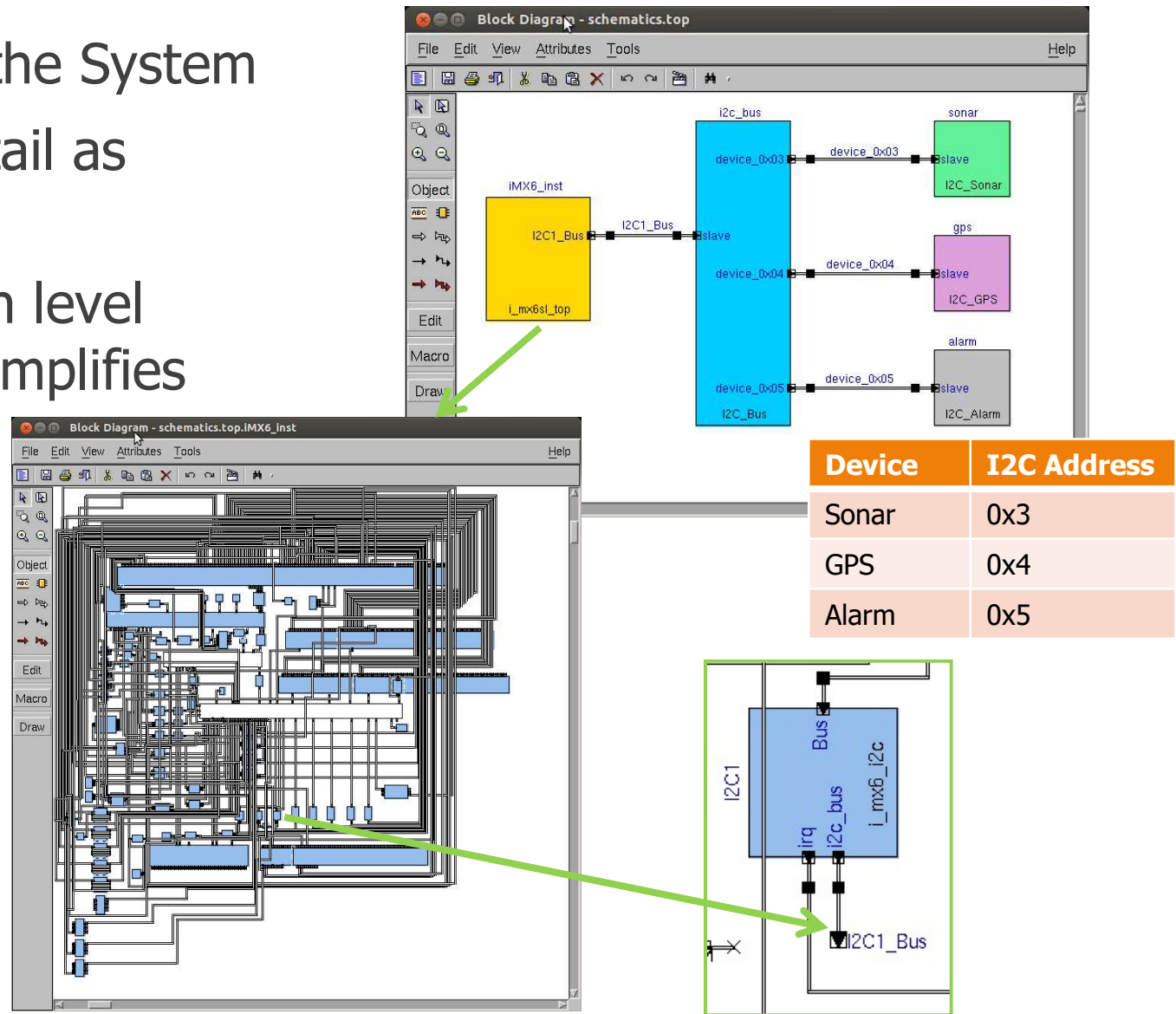
- Terminal (top right):** Shows kernel boot parameters and ELF loading progress for the top.iMX6_inst.CORTEX_A9MP.PV cores. Parameters include `call_to_default_if = 0`, `verbose_parameters = 0x1`, `dmt_enabled = 0x1`, `warning_level = IGNORE`, `slave_pipeline_length = 0x2`, `Memory_outstanding = 0x1`, `Memory_read_data_queue_size = 0`, `Memory_write_data_queue_size = 0`, `touch_if_pipeline_length = 0x2`, `frame_refresh_rate = 10 ms`, `connected = 0x1`, `baremetal = 0`, and `NU = 0`. It also shows the environment: "Awaiting connection from GUI on port 5005" and "Environment: GUI connected, awaiting commands".
- UART2: Mentor Linux (middle left):** Shows system initialization logs:

```
[ OK ] Found device /dev/ttyMX1.  
Starting WPA supplicant...  
[ OK ] Started Connection service.  
[ OK ] Reached target Remote File Systems.  
Starting Trigger Flushing of Journal to Persistent Storage...  
[ OK ] Started WPA supplicant.  
[ 8.199513] systemd-journald[140]: Received request to flush runtime journal  
from PID 1  
[ OK ] Started Trigger Flushing of Journal to Persistent Storage.  
Starting Permit User Sessions...  
[ OK ] Started Permit User Sessions.  
Starting Serial Getty on ttyMX1...  
[ OK ] Started Serial Getty on ttyMX1.  
[ OK ] Reached target login Prompts.  
[ OK ] Reached target Multi-User System.  
[ OK ] Reached target Graphical Interface.  
Starting Update UTMP about System Runlevel Changes...  
[ OK ] Started Update UTMP about System Runlevel Changes.
```
- top.iMX6_inst.IPU2.PV (bottom right):** Shows a graphical interface for a sonar device. The status is "RUNNING". The interface displays a sonar scan with a yellow and blue background. A small boat icon is visible. The data shown is:

Depth: 71
Time: 17.4
Latitude: 0
Longitude: 175

Structure

- Configure the System
- Expose detail as required
- Transaction level interface simplifies modeling



System Startup

- Image identical to physical device
- Virtual models for environment
- Boot and execution sequence matches physical device usage

```
UART1: Mentor Nucleus
Nucleus RTOS Running...

I2C Peripheral Status
-----
Sonar, Depth      : 60
GPS, Milliseconds : 42700
GPS, Latitude     : 0
GPS, Longitude    : 428
Alarm             : off

RPMSG Packets sent : 0

UART2: Mentor Linux
[ 24.222698] remoteproc0: powering up 10000000.remoteproc-test
[ 24.222758] remoteproc0: Booting fw image firmware, size 807525
[ 24.223252] I/O memory request successful. dev_name : 10000000.remoteproc-test
[ 24.223314] CPU 1 successfully started
[ 24.223361] remoteproc0: remote processor 10000000.remoteproc-test is now up
[ 25.741482] virtio_rpmsg_bus virtio0: rpmsg host is online
[ 25.741570] remoteproc0: registered virtio0 (type 7)
[ 25.742343] virtio_rpmsg_bus virtio0: creating channel rpmsg-memf-demo-channel
l addr 0x1
[ 25.747118] rpmsg_user_dev_driver rpmsg0: new channel: 0x400 -> 0x1
>>> Opening rpmsg device /dev/rpmsg...OK.
>>> Creating mutex to synchronise threads...OK.
>>> Creating UI and receive threads...OK.
>>> Running threads...

Linux Control Application
Enter command and press enter

1 - Start receiving data from Nucleus
2 - Stop receiving data from Nucleus
3 - Print current data
4 - Execute Linux Qt application
5 - Stop linux Qt application
6 - Halt Linux

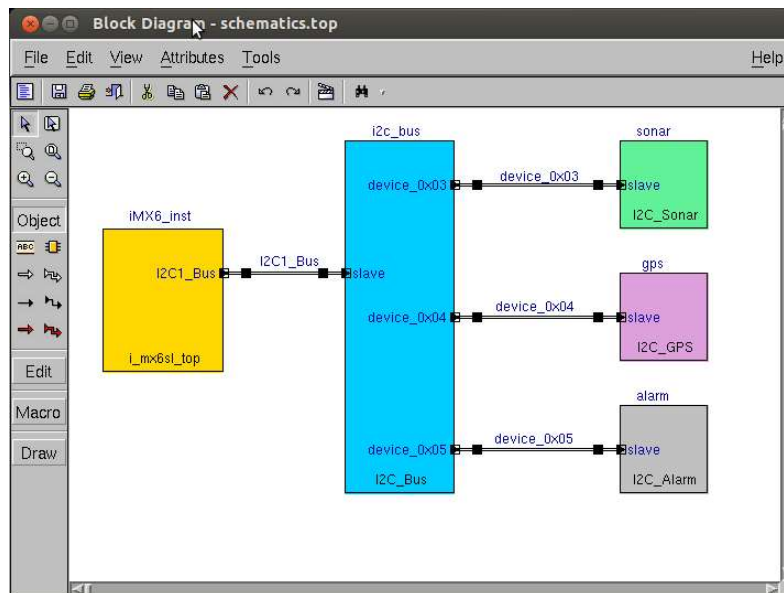
CMD>
```

```
UART2: Mentor Linux
Starting Create Volatile Files and Directories...
[ OK ] Started Load/Save Random Seed.
[ OK ] Started Create Volatile Files and Directories.
Starting Update UTMP about System Boot/Shutdown...
[ OK ] Started Update UTMP about System Boot/Shutdown.
[ OK ] Reached target System Initialization.
[ OK ] Listening on RRPCbind Server Activation Socket.
[ OK ] Listening on D-Bus System Message Bus Socket.
[ OK ] Reached target Timers.
Starting Console System Startup Logging...
[ OK ] Listening on sshd.socket.
[ OK ] Started Console System Startup Logging.
[ OK ] Reached target Sockets.
[ OK ] Reached target Basic System.
Starting Connection service...
Starting Login Service...
Starting D-Bus System Message Bus...
[ OK ] Started D-Bus System Message Bus.
[ OK ] Started Login Service.
[ OK ] Found device /dev/ttyuxcl.
Starting WPA supplicant...
[ OK ] Started WPA supplicant.
[ OK ] Started Connection service.
```

```
top:IMX6_inst.IPU2.PV
Environment: Awaiting connection from GUI on port 5085
Environment: GUI connected, awaiting commands
top:IMX6_inst.CORTEX_A9MP.PV.cpu0.core: Loading ELF: ../sw/fsbl/boot.elf
ELF section: start=0, end=0x19b
top:IMX6_inst.CORTEX_A9MP.PV.cpu0.core: Loading ELF: ../sw/linux/bootwrapper/lin
ux-imx6.elf
ELF section: start=0x20800000, end=0x20db707f
ELF section: start=0x30000000, end=0x30006910
SMP Cluster: top:IMX6_inst.CORTEX_A9MP.PV (4 cores)
```

Environment Interaction

- Environment models provide realistic input
- UI models provide output
- Enable dynamic user interaction
- Console interface for dev flow



```
UART1: Mentor Nucleus
Nucleus RTOS Running...
I2C Peripheral Status
~~~~~
Sonar, Depth      : 21
GPS, Milliseconds : 301700
GPS, Latitude     : 0
GPS, Longitude    : 3018
Alarm             : >>> RAISED <<<
RPMMSG Packets sent : 0
```

I2C Devices - Sonar, GPS, & Alarm

RUNNING

Depth: 17
Time: 302.2
Latitude: 0
Longitude: 3023



Running

- Command interface identical to physical device
- Data interaction corresponds to real world results
- Easily create complex scenarios

```
Linux Control Applicaton
Enter command and press enter

1 - Start receiving data from Nucleus
2 - Stop receiving data from Nucleus
3 - Print current data
4 - Execute Linux Qt application
5 - Stop Linux Qt application
6 - Halt Linux

CMD>1
Nucleus RTOS Running...

I2C Peripheral Status
*****
Sonar, Depth      : 121
GPS, Milliseconds : 1297100
GPS, Latitude     : 0
GPS, Longitude    : 12972
Alarm             : off

RPMSG Packets sent : 92
```

DEPTH: 99

TIME & POSITION
Time: 1306.6
Latitude: 0
Longitude: 13067

I2C Devices - Sonar, GPS, & Alarm

STOPPED

Depth: 119
Time: 1482.2
Latitude: 0
Longitude: 14800

Debugging the Virtual Platform

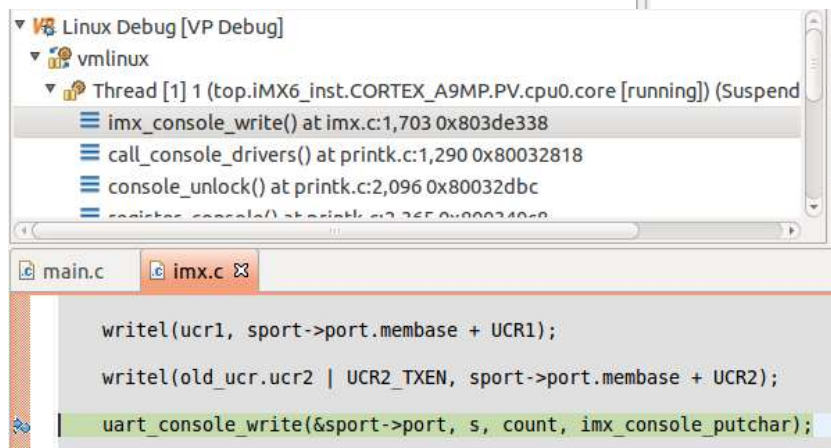
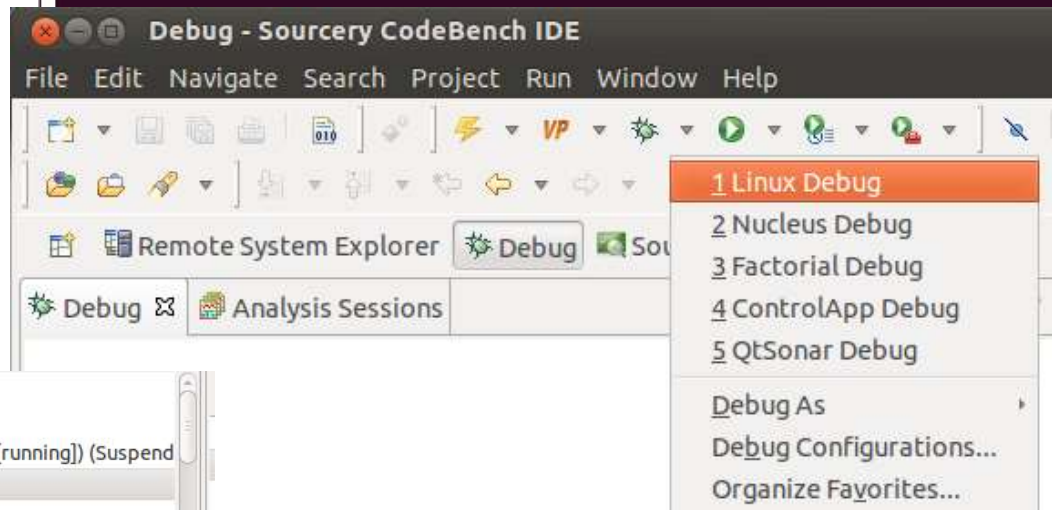
The screenshot displays the Sourcery Analyzer IDE interface for debugging a virtual platform. The interface is divided into several key sections:

- Terminal (Left):** Shows the execution of a demo channel, login to mx6q, and the start of the Nucleus RTOS. It also displays I2C Peripheral Status (Sonar Depth: 109, GPS Milliseconds: 32100, etc.) and RPMSG Packets sent (0).
- Debug Console (Right):** Shows the current thread [1] running in the Nucleus Debug (VP Debug) environment. The stack trace includes `nucleus_printInfo()` at `memf_remote.c:205`, `nucleus_Info_Task_Entry()` at `memf_remote.c:228`, and `TCC_Task_Shell()` at `tcc_common.c:703`.
- Code Editor (Bottom):** Displays the source code for `nucleus_printInfo()`, which prints various I2C peripheral status values like Sonar Depth, GPS, and Alarm.
- Status Bar (Bottom Right):** Indicates "Launching QtSonar Debug: (77%)".
- Visualizer (Bottom Center):** A small window titled "I2C Devices - Sonar, GPS, & Alarm" showing a "RUNNING" status and a sonar plot with a boat icon. The plot shows a depth of 109, time of 32.8, latitude of 0, and longitude of 327.

Linux Kernel Debug

- GDB Server connection per core
- Connection independent of HW platform
- Step through driver code

```
Terminal
Environment: Awaiting connection from GUI on port 5005
Environment: GUI connected, awaiting commands
top.iMX6_inst.CORTEX_A9MP.PV.cpu0.core: Loading ELF: ../sw/fsbl/boot.elf
ELF section: start=0, end=0x19b
top.iMX6_inst.CORTEX_A9MP.PV.cpu0.core: Loading ELF: ../sw/linux/bootwrappe
ux-imx6.elf
ELF section: start=0x20800000, end=0x20db707f
ELF section: start=0x30000000, end=0x30006910
gdbstub server is ready for connection on port '2345'
```



```
UART2: Mentor Linux
[ 0.000000] Booting Linux on physical CPU 0x0
```


Visibility

- View
 - SW Variables
 - CPU Registers
 - HW Peripheral registers

The screenshot displays a debugger's 'Registers' window. On the left, a tree view shows the 'CP15_SCTLR' register selected, with its value '0xc51078' highlighted in yellow. The main pane shows a list of hardware registers under the 'HwRegs' category, including 'IPU1_CONF' through 'IPU1_CH_BUF1_RDY0'. A secondary window in the foreground shows a list of peripheral variables under the 'IPU2' category, including 'IPU1_CONF' through 'IPU1_CH_BUF1_RDY0'.

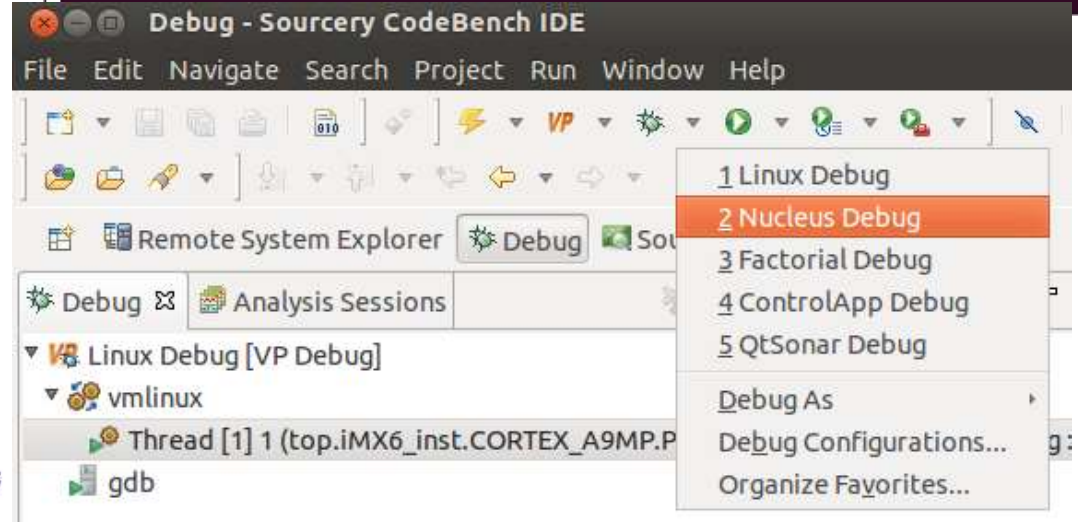
Name	Value
General Registers	
CP15	{...}
System	{...}
HwVars	{...}
HwRegs	{...}
top	{...}
iMX6_inst	{...}
uSDHC1	{...}
uSDHC4	{...}
L2C_PL310	{...}
IPU2	
(x)= IPU1_CONF	0x00000660
(x)= IPU1_FS_DISP_FLOW1	0x00000000
(x)= IPU1_DISP_GEN	0x01600000
(x)= IPU1_MEM_RST	0x007fffff
(x)= IPU1_PM	0x08100810
(x)= IPU1_GPR	0x00000000
(x)= IPU1_CH_BUF0_RDY0	0x00800000
(x)= IPU1_CH_BUF1_RDY0	0x00000000

Nucleus Debug

- Core specific debug connection
- Debug and trace specific to RTOS
- Deterministic execution in debug or free running

```
static VOID nucleus_printInfo() {  
    int r;  
    printf("Nucleus RTOS Running...\r\n");  
    printf("\r\n");  
    printf("I2C Peripheral Status\r\n");  
    printf("~~~~~\r\n");  
    printf(" Sonar, Depth : %" PRIu32 "\r\n", dp.depth);  
    printf(" GPS, Milliseconds : %" PRIu32 "\r\n", dp.timeval);  
    printf(" GPS, Latitude : %" PRIu32 "\r\n", dp.latitude);  
    printf(" GPS, Longitude : %" PRIu32 "\r\n", dp.longitude);  
    if (lastAlarm == 0x1) {  
        printf(" Alarm : >>> RAISED <<<\r\n");  
    } else {  
    }  
}
```

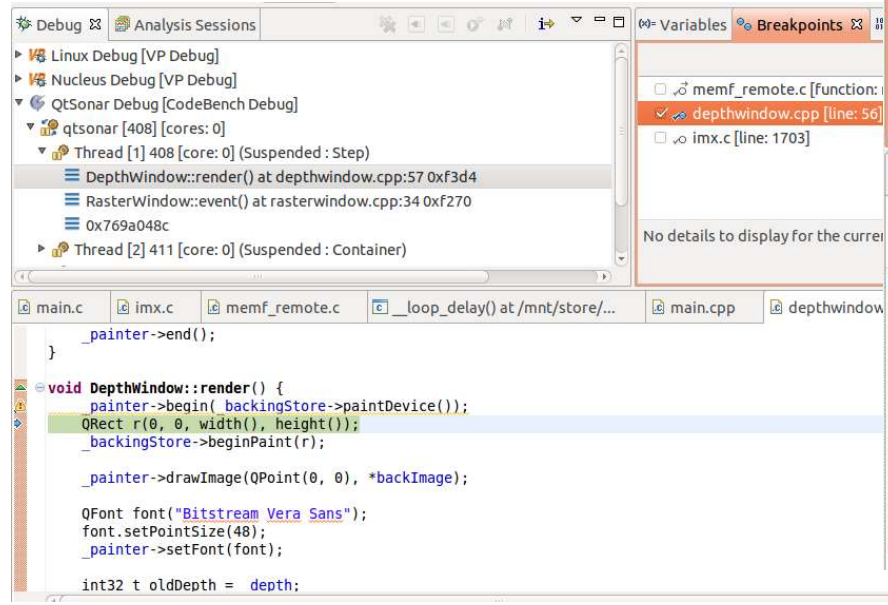
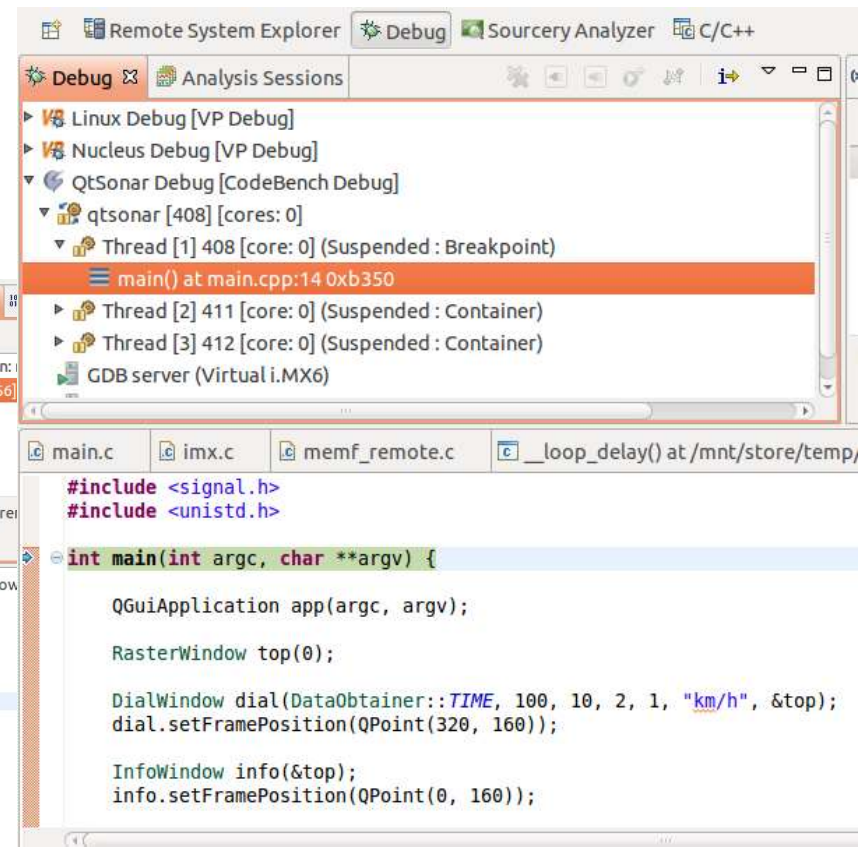
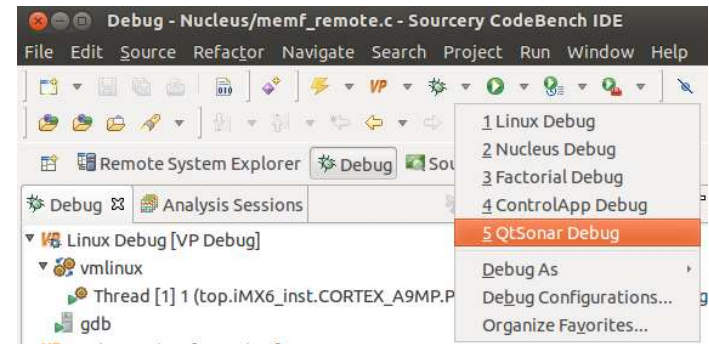
```
Terminal  
top.iMX6_inst.CORTEX_A9MP.PV.cpu0.core: Loading ELF: ../sw/fsbl/boot.elf  
ELF section: start=0, end=0x19b  
top.iMX6_inst.CORTEX_A9MP.PV.cpu0.core: Loading ELF: ../sw/linux/bootwrappe  
ux-imx6.elf  
ELF section: start=0x20800000, end=0x20db707f  
ELF section: start=0x30000000, end=0x30006910  
gdbstub server is ready for connection on port '2345'  
top.iMX6_inst.CORTEX_A9MP.PV.cpu0.core: Debugger is attached.  
gdbstub server is ready for connection on port '2346'
```



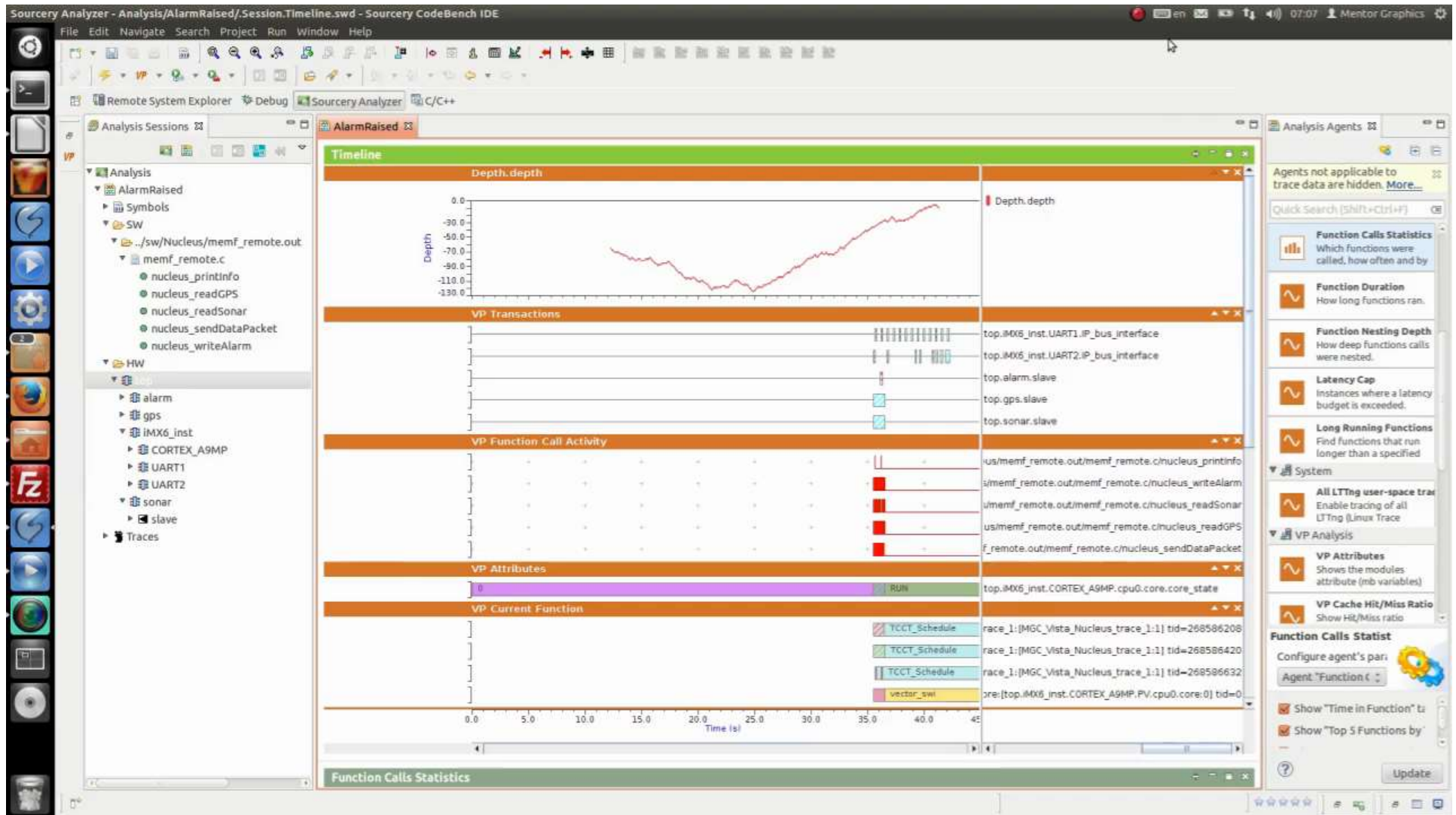
```
UART1: Mentor Nucleus  
Nucleus RTOS Running...  
I2C Peripheral Status  
~~~~~  
|
```

Application Debug

- Launch application via Ethernet
 - Full network interface
- Multi process, OS, core, aware



Analyzing the Virtual Platform



Record

- Dynamic trace control

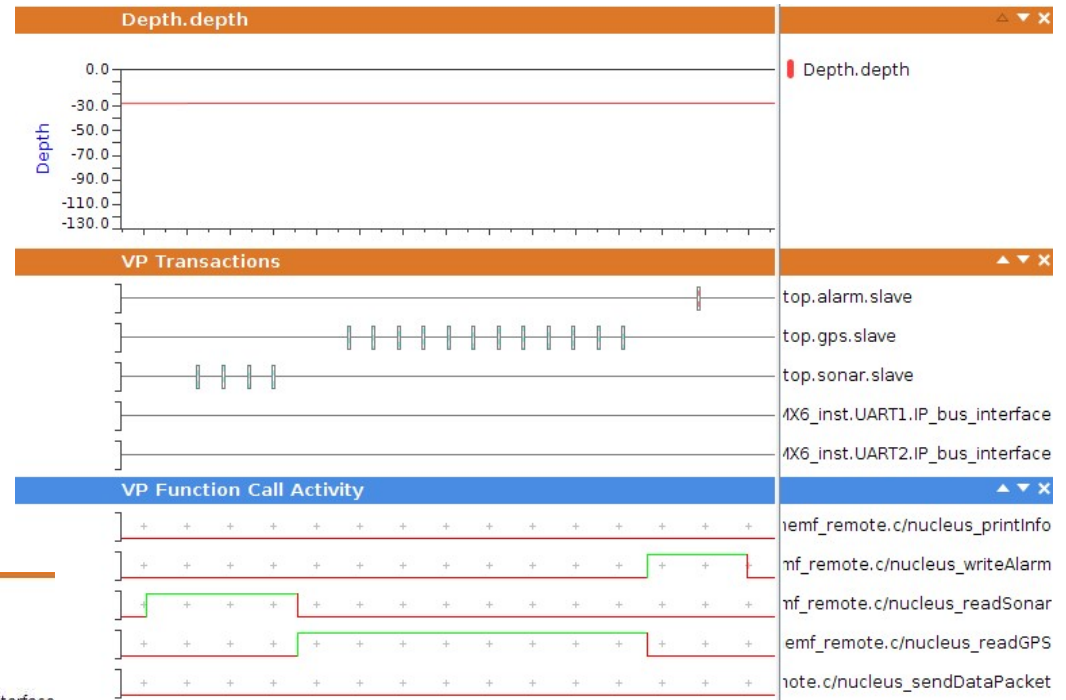
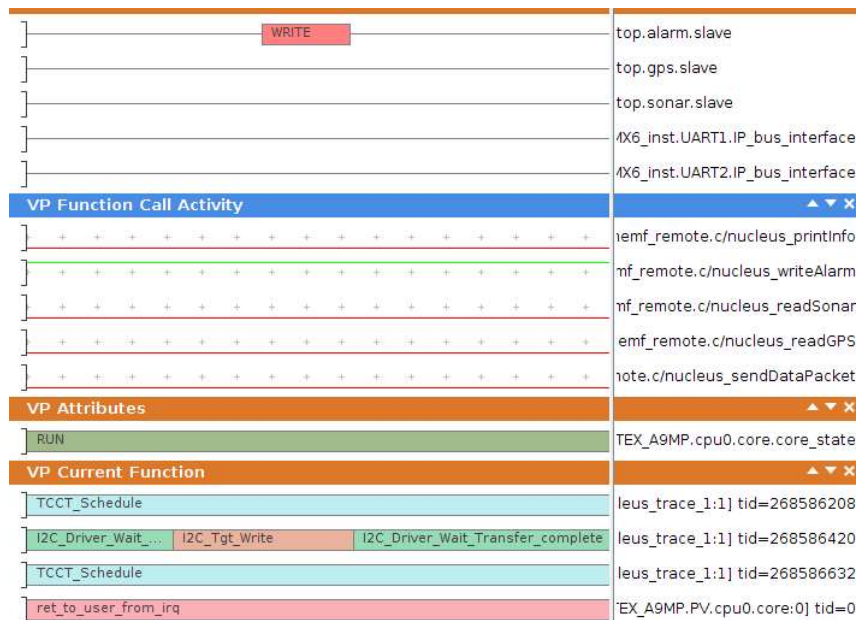
- SW Variables
- HW Transactions
- Function calls

```
#-----  
# Tracing in Linux  
  
select_core "top.imx6_inst.CORTEX_A9MP.PV.cpu0.core"  
  
trace_linux ../sw/linux/mel/vmlinux -libc ../sw/linux/mel/libc-2.20.so  
  
trace_function_calls -kind eff -tag LINUX_FUNCTION_CALLS -disabled  
trace_current_function -kind eff -tag LINUX_CURRENT_FUNCTION -disabled  
  
trace_linux_process controlapp {  
  add_symbol_file ../sw/ControlApp/controlapp  
  
  trace_function_calls -kind eff -tag CONTROL_FUNCTION_CALLS  
  
  insert_tracepoint controlapp_tp1 -at-function-entry main -do-tcl {  
    puts "NIT >>> Control App, main function entered"  
  }  
  
  insert_tracepoint controlapp_tp1 -at-function-entry startReceiving -do-tcl {  
    puts "NIT >>> Control App, startReceiving - ENABLING TRACING"  
    enable_tag HW_LINUX_CORE  
    enable_tag HW_I2C_SONAR  
    enable_tag HW_I2C_GPS  
    enable_tag HW_I2C_ALARM  
    enable_tag HW_UART1  
    enable_tag HW_UART2  
    enable_tag NUCLEUS_FUNCTION_CALLS  
    enable_tag NUCLEUS_FUNCTION_ACTIVITY  
    enable_tag NUCLEUS_CURRENT_FUNCTION  
    enable_tag LINUX_FUNCTION_CALLS  
    enable_tag LINUX_CURRENT_FUNCTION  
    enable_tag QT_FUNCTION_CALLS  
  }  
  
  insert_tracepoint controlapp_tp2 -at-function-entry startReceiving -do-row {  
    printf("NIT >>> Control App, DYNAMIC timing mode\n");  
    set_parameter("top.imx6_inst.I2C1.lt_policy_modeling", "DYNAMIC");  
    set_parameter("top.sonar.lt_policy_modeling", "DYNAMIC");  
    set_parameter("top.gps.lt_policy_modeling", "DYNAMIC");  
    set_parameter("top.alarm.lt_policy_modeling", "DYNAMIC");  
  }  
}
```



Interact

- Trace detailed register activity
- Correlate to SW



- Deterministic control of time

Visualize

- System Status
- RTOS Process activity
- Linux applications
 - Control App
 - Qt App



Statistics

- Bare Metal
- Core Activity
- Linux Process Activity

Function Calls Statistics			
Time in Function			
Linux-top.iMX6_inst.CORTEX_A9MP.PV.cpu0.core:[controlapp:389]			
Function Name	Time with Children	↑ Time without Children	Rel. Time without Children [%]
ui_dispatcher	21.033812368 s	19.982039509 s	43.38%
checkForReceiveFlag	17.313662268 s	17.313662268 s	37.58%
receive_thread_entry	21.729472929 s	3.325598599 s	7.22%
openRPMMSGDevice	2.518146363 s	2.518146363 s	5.47%
pull_datapacket	1.085298499 s	1.085298499 s	2.36%
startReceiving	1.000175256 s	1.000175256 s	2.17%
ui_thread_entry	21.643493013 s	609.680645 ms	1.32%
clearFB	87.683636 ms	87.683636 ms	0.19%
startRPMMSG	77.938912 ms	77.938912 ms	0.17%
startRemoteProc	53.387208 ms	53.387208 ms	0.12%
stopQt	50.991087 ms	6.423886 ms	0.01%
request_datapacket	4.913563 ms	4.913563 ms	0.01%
executeQt	606.516 µs	606.516 µs	0.00%
main	2.693019260 s	371.750 µs	0.00%
createSharedMem	58.592 µs	58.592 µs	0.00%
__libc_csu_init	248.0 ns	248.0 ns	0.00%
halt	0 s	0 s	0.00%
Totals		46.065985450 s	100.00%

Linux-top.iMX6_inst.CORTEX_A9MP.PV.cpu0.core:[qtsonar:395]			
Function Name	Time with Children	↑ Time without Children	Rel. Time without Children [%]
RasterWindow::event	1.308470189 s	651.425671 ms	65.99%
0x276D	134.505530 ms	134.504940 ms	13.63%
InfoWindow::render	1.217811228 s	101.949937 ms	10.33%
0x207E	35.214220 ms	35.213952 ms	3.57%

Questions:

Thank You!

