Application Software Pack: Power Optimized Wearables User Interface for the i.MX RT595

Lab Hand Out - Revision 1

NXP Semiconductors
www.nxp.com

Contents

1 Lab Overview

The application in this lab showcases the power optimization techniques for a Voice UI use case. This lab covers how to run, customize, and profile this application. Lastly, it explains how to add power optimizations in your own application.

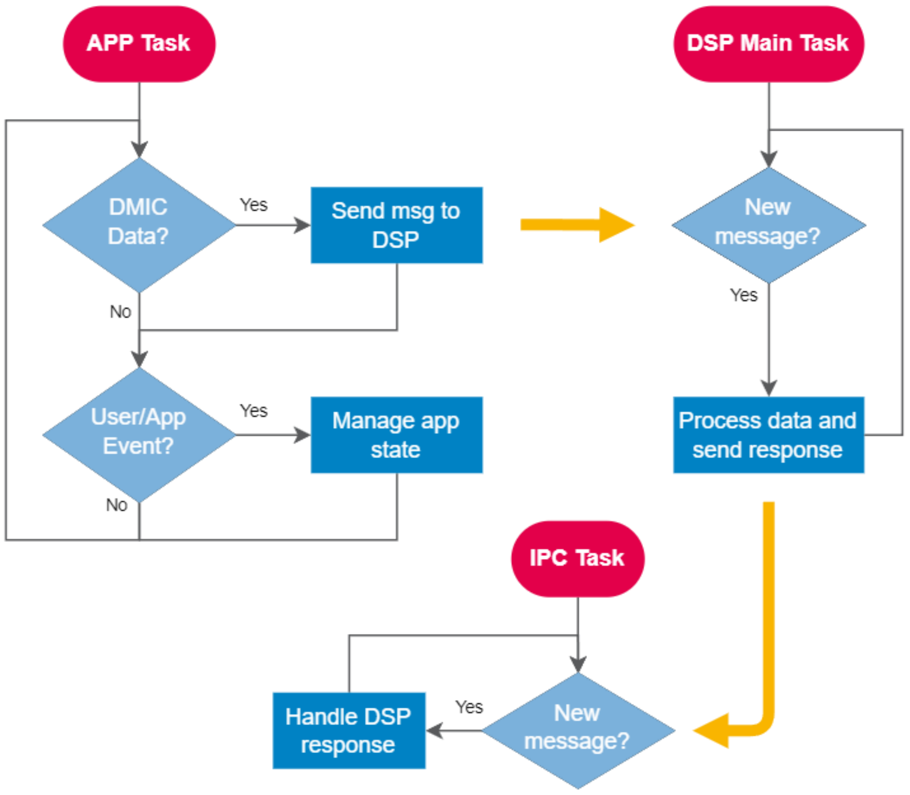
1.1 Application Description

The application first initializes the *main_clk* to use the 96 MHz FRO and sets the CPU divider to 8, meaning that the core frequency is set to 12 MHz. Next, it initializes the common peripherals used by all the application states, copies the DSP image from external Flash to SRAM and powers down the unused peripherals, including the Flash and runs the rest of the application from SRAM.

After initializing the necessary hardware, it launches the following tasks:

1. **[CM33] APP_Task:** Main application task to handle the user/application events and manage the current application state.
2. **[CM33] APP_DSP_IPC_Task:** Inter-processor communication (IPC) task to handle the DSP response messages.
3. **[CM33] APP_Idle:** Function called from the RTOS idle task to handle the power mode entry of the device.
4. **[DSP] DSP_Main:** DSP main task to handle the CM33 messages.

   The general flow of the tasks is shown below.



2 Equipment

The following equipment is needed for this lab:

- MIMXRT595-EVK
- Micro-B USB cable
- MCUXpresso IDE V11.8.0+
- TeraTerm (or any other terminal emulator)
- Power measurement tool (optional)

3 Resources

The following are helpful resources for this lab:

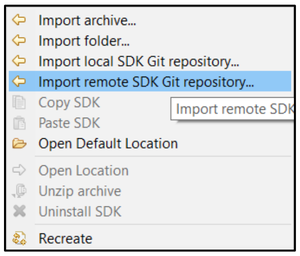
- Software Pack Repository
- Application Note AN13758

4 Import the App Software Pack Into MCUXpresso IDE

There are two methods to get the application software pack. This section will cover both options, but only one needs to be done. It is recommended to use the first option as it is more straight forward.

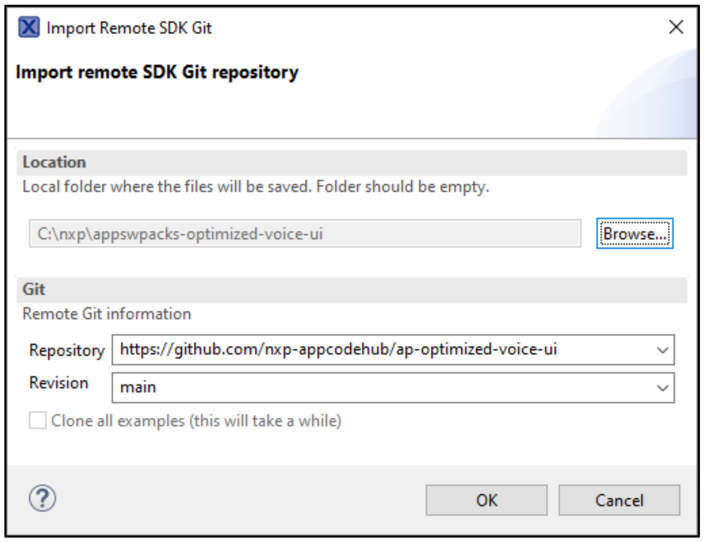
4.1 Option #1: Get the App Software Pack with MCUXpresso IDE

1. Open MCUXpresso IDE and select a workspace location in an empty directory.
2. Right click in the blank area of the **Installed SDKs** panel at the bottom and select **Import remote SDK Git repository…**



3. In the dialog box that comes up:
   a) In the **Location** field, click on the Browse button and select or create an empty directory for the application software pack to be downloaded to. Make note of this location as it'll be used throughout this lab.
   b) In the **Repository** field put: **https://github.com/nxp-appcodehub/ap-optimized-voice-ui**
   c) In the **Revision** field put: **main**

   Then hit **OK** to download the application software pack.


Import Remote SDK Git

Import remote SDK Git repository

Location

Local folder where the files will be saved. Folder should be empty.

C:\nxp\appswpacks-optimized-voice-ui

Git

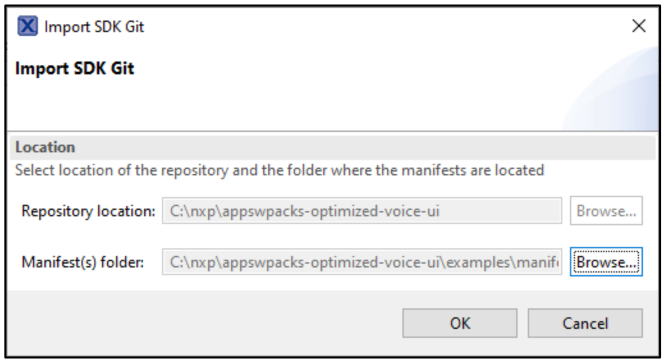
Remote Git information

Repository: https://github.com/nxp-appcodehub/ap-optimized-voice-ui

Revision: main

Clone all examples (this will take a while)

4. In the dialog box that comes up:
   a) In the **Manifest(s) folder:** field, select the **manifests** folder under the **examples** folder.


Import SDK Git

Import SDK Git

Location

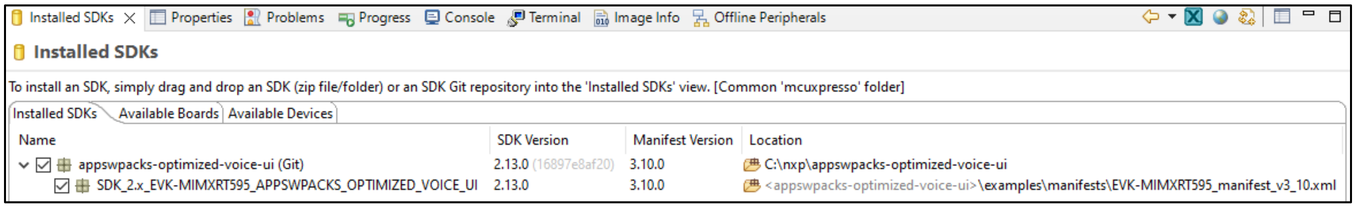
Select location of the repository and the folder where the manifests are located

Repository location: C:\nxp\appswpacks-optimized-voice-ui

Manifest(s) folder: C:\nxp\appswpacks-optimized-voice-ui\examples\manif

Click **OK** to continue importing the application software pack.

5. Once imported the Installed SDKs tab will look like this:


Installed SDKs

To install an SDK, simply drag and drop an SDK (zip file/folder) or an SDK Git repository into the 'Installed SDKs' view. [Common 'mcuxpresso' folder]

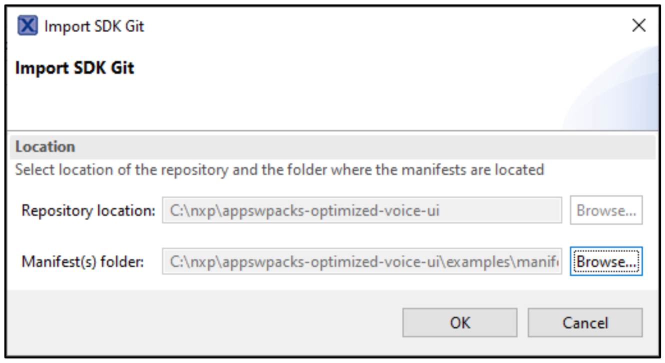
Name	SDK Version	Manifest Version	Location
appswpacks-optimized-voice-ui (Git)	2.13.0 (16897e8af20)	3.10.0	C:\nxp\appswpacks-optimized-voice-ui
SDK_2.x_EVK-MIMXRT595_APPSWPACKS_OPTIMIZED_VOICE_UI	2.13.0	3.10.0	<appswpacks-optimized-voice-ui>\examples\manifests\EVK-MIMXRT595_manifest_v3_10.xml

4.2 Option #2: Use the Command Line

1. Open up a Windows command line and execute the following lines:

```
west init -m https://github.com/nxp-appcodehub/ap-optimized-voice-ui --mr main appswpacks_optmized_voice_ui
cd appswpacks_optimized_voice_ui
west update
```

2. Next open MCUXpresso IDE and select a workspace location in an empty directory.
6. Drag-and-drop the \appswpacks_optimized_voice_ui directory that was created in the previous step into the **Installed SDKs** window, located on a tab at the bottom of the screen named “Installed SDKs”.
7. In the dialog box that comes up:
   a) In the **Manifest(s) folder:** field, select the **manifests** folder under the **examples** folder.


Import SDK Git

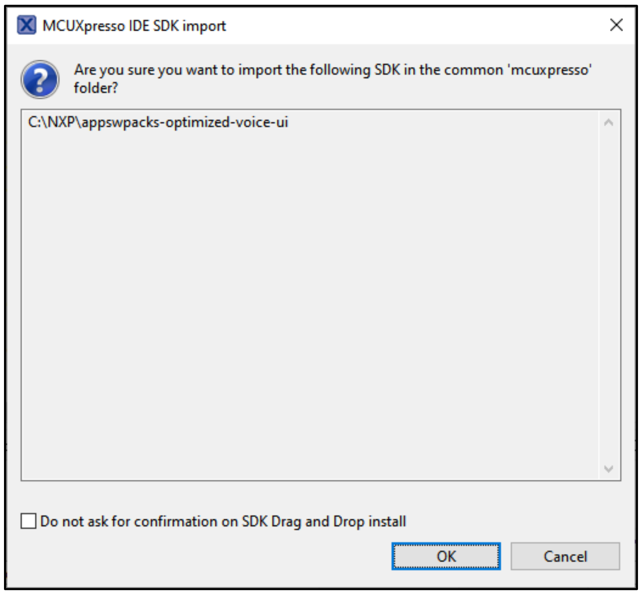
Import SDK Git

Location
Select location of the repository and the folder where the manifests are located

Repository location: C:\nxp\appswpacks-optimized-voice-ui
Manifest(s) folder: C:\nxp\appswpacks-optimized-voice-ui\examples\manif

Click **OK** to continue importing the application software pack.

3. Click **OK** to confirm.

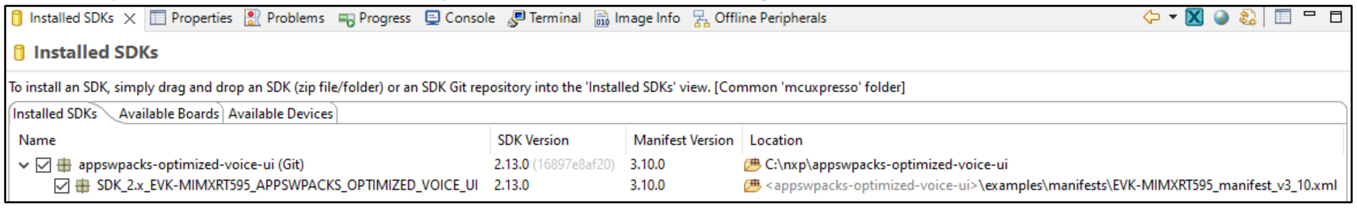

MCUXpresso IDE SDK import

Are you sure you want to import the following SDK in the common 'mcuxpresso' folder?

C:\NXP\appswpacks-optimized-voice-ui

☐ Do not ask for confirmation on SDK Drag and Drop install

4. Once imported, the Installed SDK panel will look something like this:



Installed SDKs

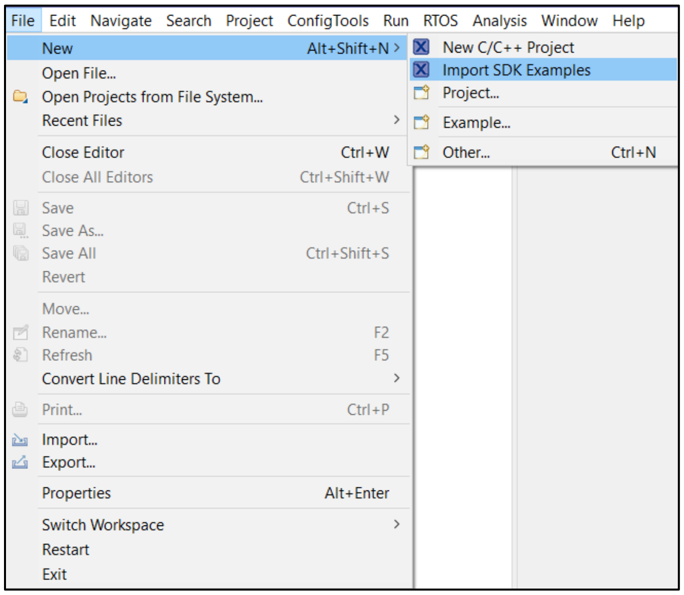
To install an SDK, simply drag and drop an SDK (zip file/folder) or an SDK Git repository into the 'Installed SDKs' view. [Common 'mcuxpresso' folder]

Name	SDK Version	Manifest Version	Location
☑ appswpacks-optimized-voice-ui (Git)	2.13.0 (16897e8af20)	3.10.0	C:\nxp\appswpacks-optimized-voice-ui
☑ SDK_2.x_EVK-MIMXRT595_APPSWPACKS_OPTIMIZED_VOICE_UI	2.13.0	3.10.0	<appswpacks-optimized-voice-ui>\examples\manifests\EVK-MIMXRT595_manifest_v3_10.xml

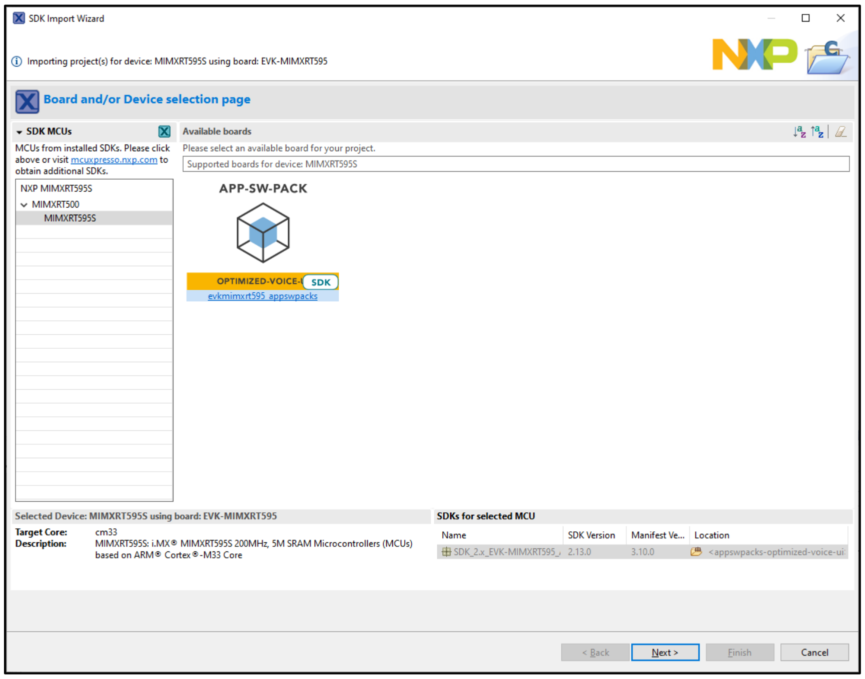
4.3 Importing the Project Into Your Workspace

Once you finish importing the Software Pack into the Installed SDKs, it is time to import the project into your workspace.

1. Go to **File -> New -> Import SDK Examples**



2. Once the SDK Import Wizard pops up, select **evkmimxrt595-appswpacks** and click **Next**


SDK Import Wizard

Importing project(s) for device: MIMXRT595S using board: EVK-MIMXRT595

Board and/or Device selection page

SDK MCUs

MCUs from installed SDKs. Please click above or visit mcuxpresso.nxp.com to obtain additional SDKs.

Available boards

Please select an available board for your project.

Supported boards for device: MIMXRT595S

Selected Device: MIMXRT595S using board: EVK-MIMXRT595

Target Core: cm33

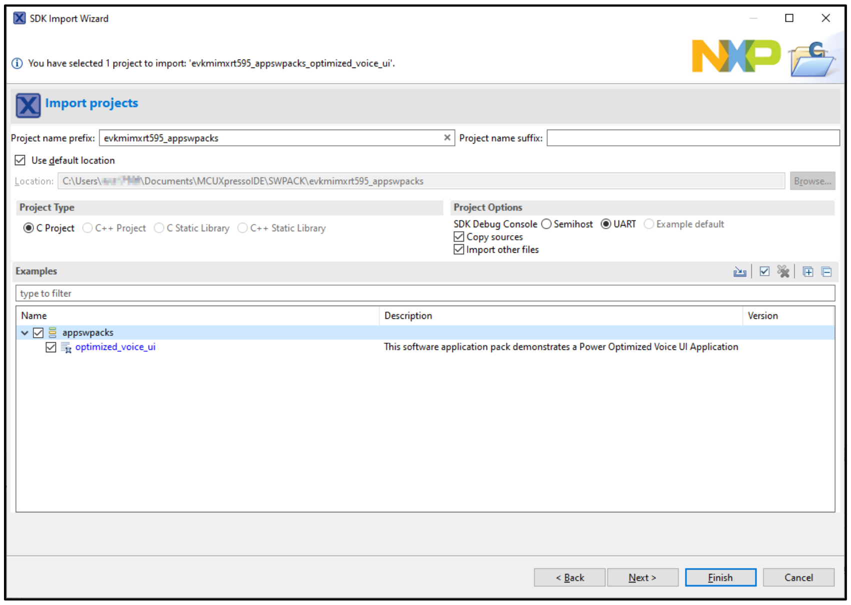
Description: MIMXRT595S: i.MX® MIMXRT595S 200MHz, 5M SRAM Microcontrollers (MCUs) based on ARM® Cortex®-M33 Core

SDKs for selected MCU

Name	SDK Version	Manifest Ve...	Location
SDK_2.x_EVK-MIMXRT595_	2.13.0	3.10.0	<appswpacks-optimized-voice-ui

3. Check the **appswpacks** checkbox.
4. Check or Uncheck the **Copy sources** checkbox.
    a. If checked, MCUXpresso will make copies of all project files and place them in your selected workspace directory.

b. If unchecked, MCUXpresso will link the project files to the files in the repository that was cloned in the previous sections (i.e. appswpacks_optimized_voice_ui)



SDK Import Wizard

You have selected 1 project to import: 'evkmimxrt595_appswpacks_optimized_voice_ui'.

Import projects

Project name prefix: evkmimxrt595_appswpacks Project name suffix:

☑ Use default location

Location: C:\Users\[redacted]\Documents\MCUXpressoIDE\SWPACK\evkmimxrt595_appswpacks Browse...

Project Type: [✓ C Project] C++ Project C Static Library C++ Static Library

Project Options: SDK Debug Console Semihost [✓ UART] Example default
☑ Copy sources
☑ Import other files

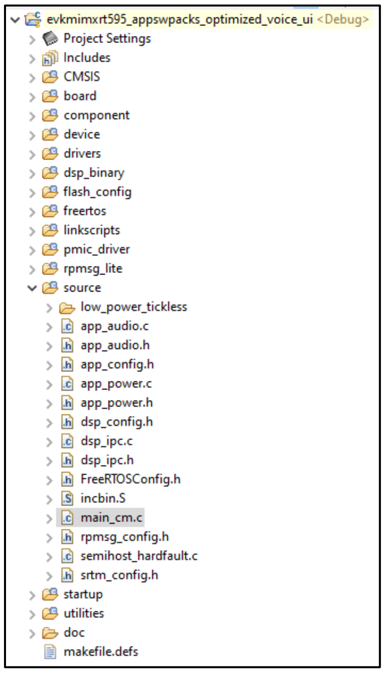
Examples

type to filter

Name	Description	Version
☑ appswpacks		
☑ optimized_voice_ui	This software application pack demonstrates a Power Optimized Voice UI Application	

< Back Next > Finish Cancel

5. Click **Finish**.
6. Once done, your workspace should look like below.



```
evkmimxrt595_appswpacks_optimized_voice_ui <Debug>
  Project Settings
  Includes
  CMSIS
  board
  component
  device
  drivers
  dsp_binary
  flash_config
  freertos
  linkscripts
  pmic_driver
  rpmsg_lite
  source
    low_power_tickless
    app_audio.c
    app_audio.h
    app_config.h
    app_power.c
    app_power.h
    dsp_config.h
    dsp_ipc.c
    dsp_ipc.h
    FreeRTOSConfig.h
    incbin.S
    main_cm.c
    rpmsg_config.h
    semihost_hardfault.c
    srtm_config.h
  startup
  utilities
  doc
  makefile.defs
```

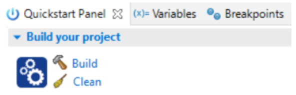
5 Running the Application

This section describes how to configure your EVK jumpers for the application and then walks you through the steps to build, flash, and run the application.

5.1 Building, Flashing, and Running

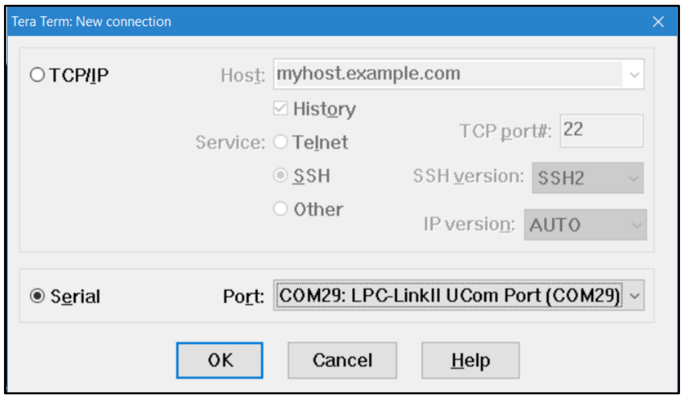
Once you finish configuring the EVK jumpers, you can build, flash, and run the application.

1. Build the project by clicking **Build** in the Quickstart Panel.



2. Once the build finishes, the output console should look like below.

3. Connect the micro-B USB cable to **J40** on the EVK.
4. Open TeraTerm or a different terminal emulator, and connect to the COM port belonging to the EVK.

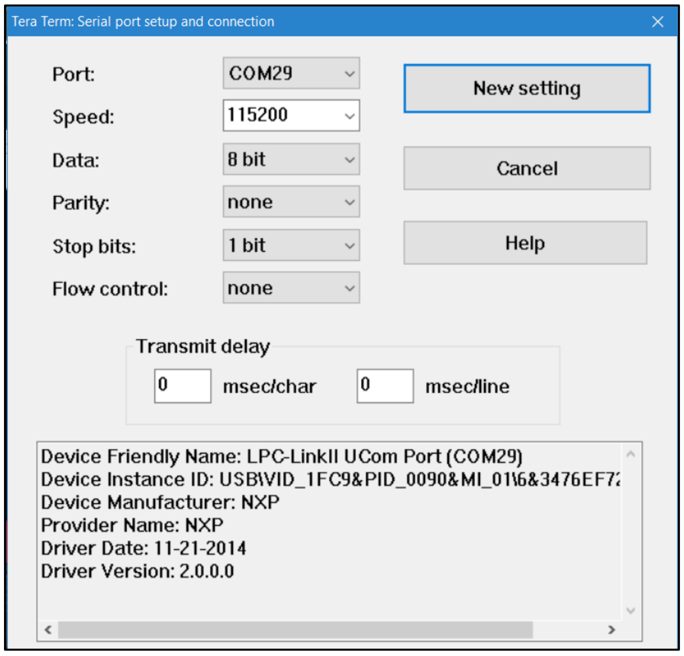


Tera Term: New connection

○ TCP/IP — Host: myhost.example.com — ☑ History — Service: ○ Telnet, ◉ SSH, ○ Other — TCP port#: 22 — SSH version: SSH2 — IP version: AUTO

◉ Serial — Port: COM29: LPC-LinkII UCom Port (COM29)

5. In TeraTerm, go to **Setup -> Serial Port** and configure the settings to use **115200 baud, 1 stop bit, no parity**.



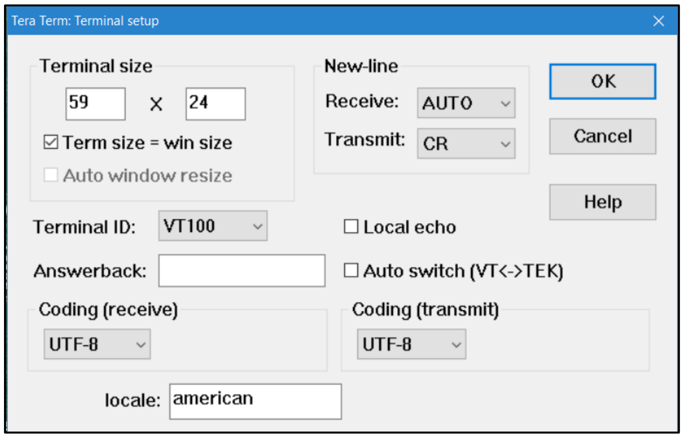
Tera Term: Serial port setup and connection

Port:	COM29
Speed:	115200
Data:	8 bit
Parity:	none
Stop bits:	1 bit
Flow control:	none

Transmit delay: 0 msec/char 0 msec/line

```
Device Friendly Name: LPC-LinkII UCom Port (COM29)
Device Instance ID: USB\VID_1FC9&PID_0090&MI_01\6&3476EF7
Device Manufacturer: NXP
Provider Name: NXP
Driver Date: 11-21-2014
Driver Version: 2.0.0.0
```

6. In TeraTerm, go to **Setup -> Terminal** and change the **New-line Receive** to **Auto** or **LF**. This is necessary for the Coremark results to print correctly.



Tera Term: Terminal setup

Terminal size: 59 x 24
☑ Term size = win size
☐ Auto window resize

New-line
Receive: AUTO
Transmit: CR

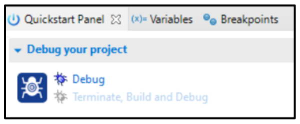
Terminal ID: VT100
Answerback:
☐ Local echo
☐ Auto switch (VT<->TEK)

Coding (receive): UTF-8
Coding (transmit): UTF-8

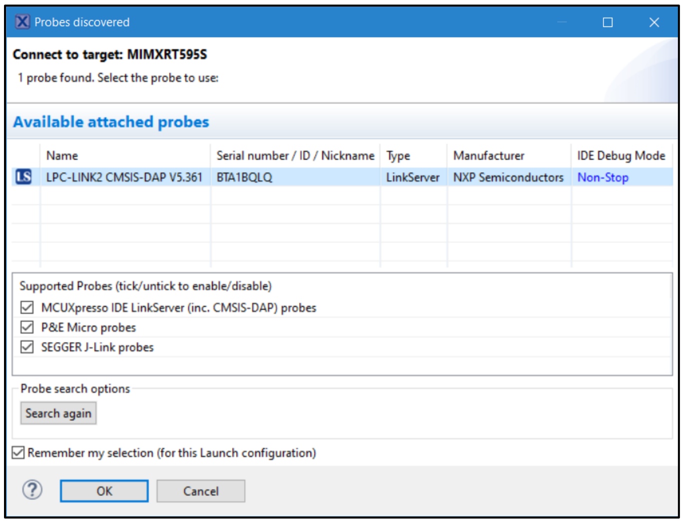
locale: american

OK | Cancel | Help

7. Debug the project by clicking on **Debug** in the Quickstart Panel.



8. It will ask what interface to use. Select the detected debug probe. The screenshot below shows the default CMSIS-DAP probe.



Probes discovered

Connect to target: MIMXRT595S

1 probe found. Select the probe to use:

Available attached probes

Name	Serial number / ID / Nickname	Type	Manufacturer	IDE Debug Mode
LPC-LINK2 CMSIS-DAP V5.361	BTA1BQLQ	LinkServer	NXP Semiconductors	Non-Stop

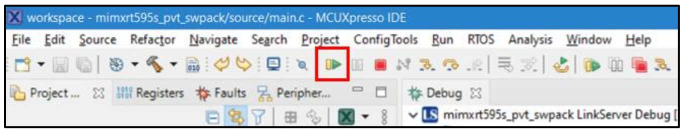
Supported Probes (tick/untick to enable/disable)
☑ MCUXpresso IDE LinkServer (inc. CMSIS-DAP) probes
☑ P&E Micro probes
☑ SEGGER J-Link probes

Probe search options
Search again

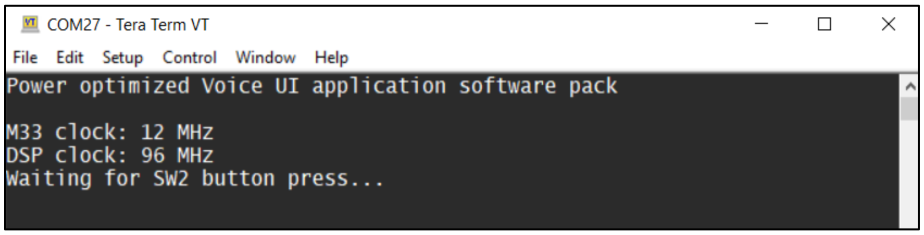
☑ Remember my selection (for this Launch configuration)

OK | Cancel

9. The debugger will download the firmware and open up the debug view. Click on the Resume button to start running.


workspace - mimxrt595s_pvt_swpack/source/main.c - MCUXpresso IDE
File Edit Source Refactor Navigate Search Project ConfigTools Run RTOS Analysis Window Help
Project ...
Registers
Faults
Peripher...
Debug
mimxrt595s_pvt_swpack LinkServer Debug

10. The following text should appear in the terminal program.



```
Power optimized Voice UI application software pack

M33 clock: 12 MHz
DSP clock: 96 MHz
Waiting for SW2 button press...
```

6 Customizing the Application

There are several ways to customize the application to help you adapt to different application requirements. The following defines are located in **app_config.h**.

6.1 Enable/Disable the User Action state

You can disable the User Action state by setting **USER_ACTION_ENABLE** to 0.

6.2 Enable/Disable the HWVAD state

You can disable the User Action state by setting **HWVAD_ENABLE** to 0.

6.3 Enable/Disable the SWVAD state

You can disable the User Action state by setting **SWVAD_ENABLE** to 0.

6.4 Enable/Disable the audio buffering in HWVAD state

You can disable the audio buffering during the HWVAD state by setting **HWVAD_BUFFERING** to 0.

6.5 DMIC Clock source

The DMIC clock source can be selected from the 1 MHz low-power oscillator (LPOSC) or from the 96 MHz free-running oscillator (FRO). The FRO is selected when **DMIC_USE_FRO** is set to 1, otherwise the LPOSC is selected.

6.6 Timeout duration

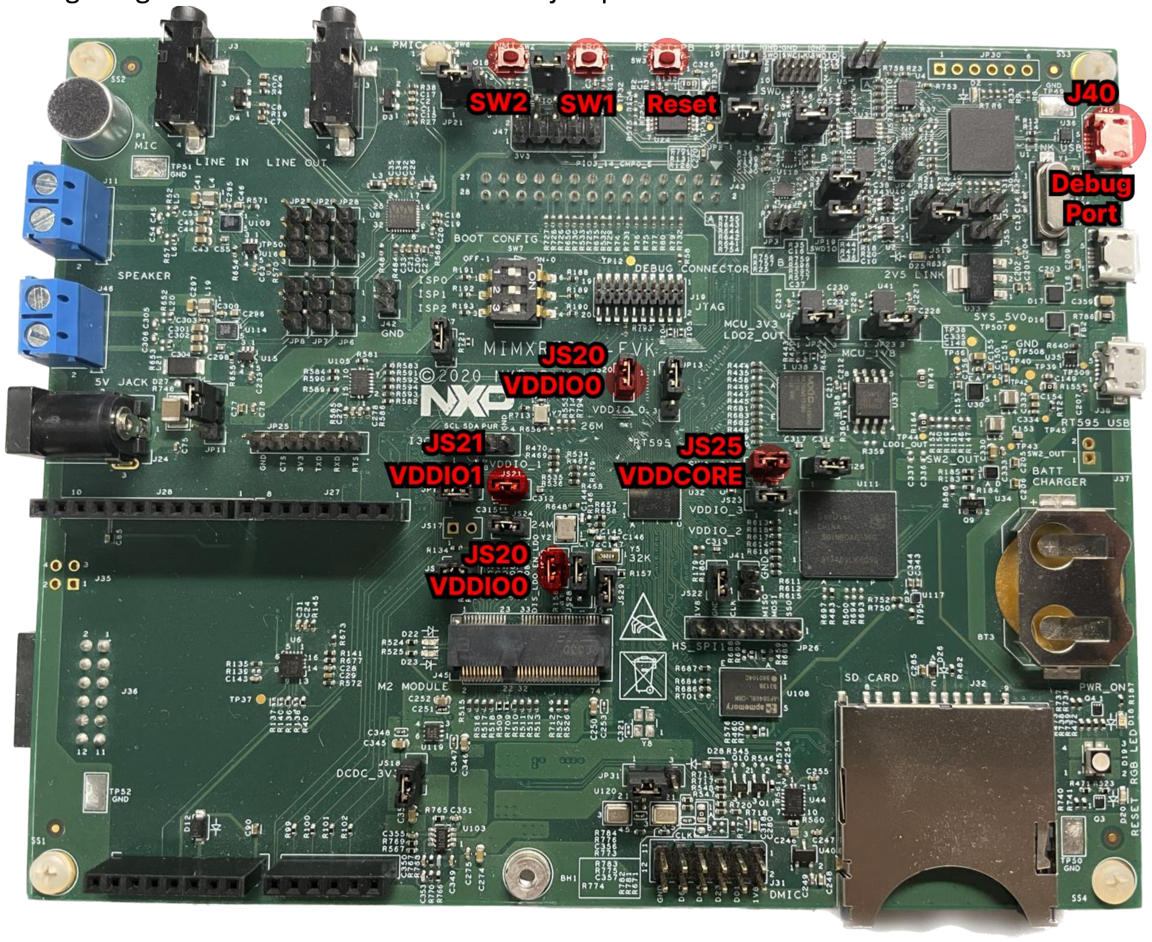
The timeout duration can be configured with the **TIMEOUT_SECS** macro definition.

7 Taking power measurements

The EVK has jumpers available to take power measurements on different power rails. The relevant power rails for this application software pack are the following:

- **JS25: PMIC_SW1_OUT** – VDDCORE
- **JS30: PMIC_SW2_OUT** - VDD1V8
- **JS20: PMIC_SW2_OUT** - VDDIO0
- **JS21: PMIC_SW2_OUT** - VDDIO1

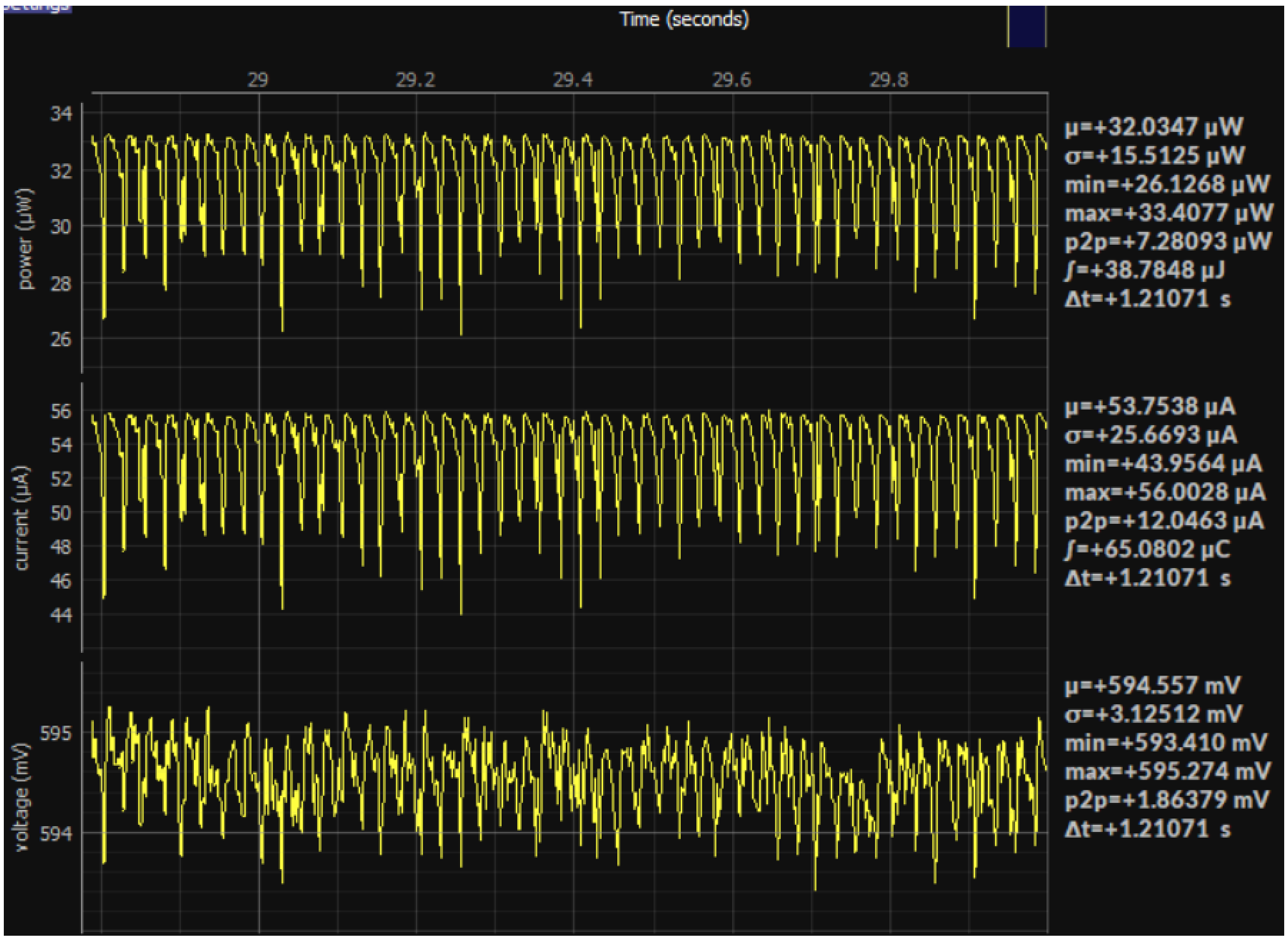
The following image shows the locations of these jumpers on the EVK:



7.1 VDDCORE Power measurement

Below are some sample captures of the power consumption from the VDDCORE power rail during the different application states.

7.1.1 User Action

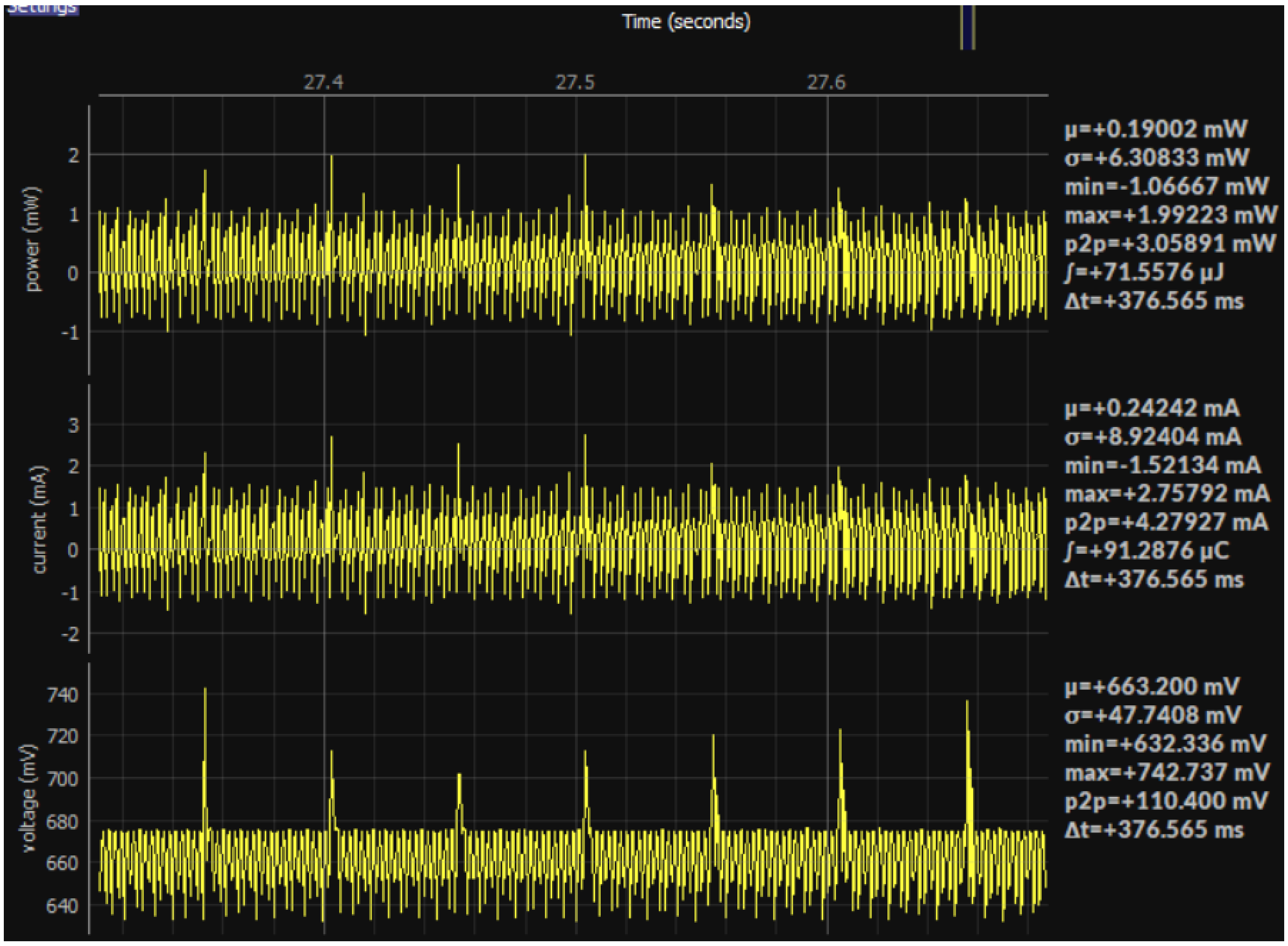


μ=+32.0347 μW
σ=+15.5125 μW
min=+26.1268 μW
max=+33.4077 μW
p2p=+7.28093 μW
∫=+38.7848 μJ
Δt=+1.21071 s

μ=+53.7538 μA
σ=+25.6693 μA
min=+43.9564 μA
max=+56.0028 μA
p2p=+12.0463 μA
∫=+65.0802 μC
Δt=+1.21071 s

μ=+594.557 mV
σ=+3.12512 mV
min=+593.410 mV
max=+595.274 mV
p2p=+1.86379 mV
Δt=+1.21071 s

7.1.2 HW VAD

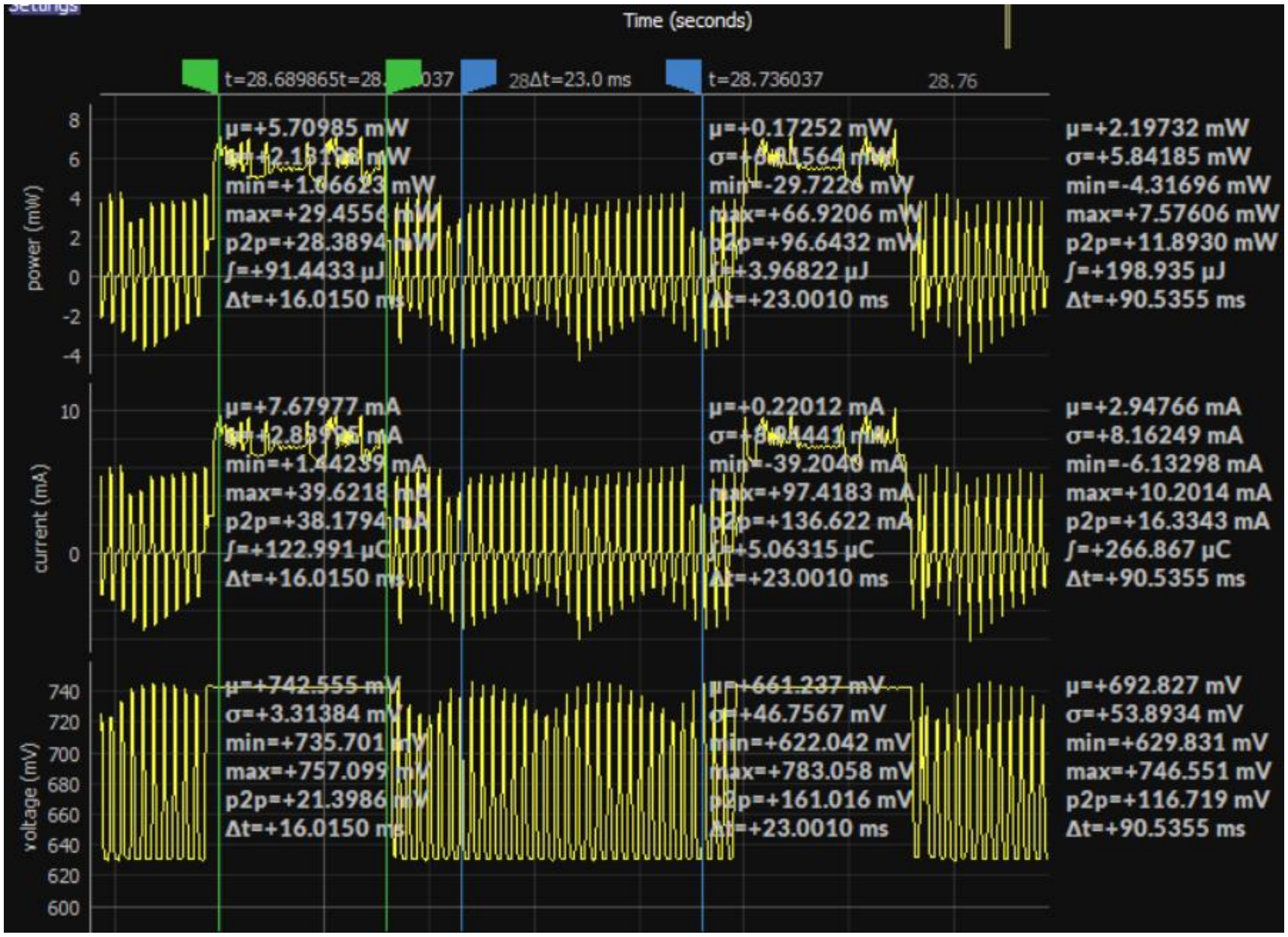


μ=+0.19002 mW
σ=+6.30833 mW
min=-1.06667 mW
max=+1.99223 mW
p2p=+3.05891 mW
∫=+71.5576 μJ
Δt=+376.565 ms

μ=+0.24242 mA
σ=+8.92404 mA
min=-1.52134 mA
max=+2.75792 mA
p2p=+4.27927 mA
∫=+91.2876 μC
Δt=+376.565 ms

μ=+663.200 mV
σ=+47.7408 mV
min=+632.336 mV
max=+742.737 mV
p2p=+110.400 mV
Δt=+376.565 ms

7.1.3 Processing



```
μ=+5.70985 mW        μ=+0.17252 mW        μ=+2.19732 mW
σ=+2.13198 mW        σ=+6.01564 mW        σ=+5.84185 mW
min=+1.66623 mW      min=-29.7226 mW      min=-4.31696 mW
max=+29.4556 mW      max=+66.9206 mW      max=+7.57606 mW
p2p=+28.3894 mW      p2p=+96.6432 mW      p2p=+11.8930 mW
∫=+91.4433 μJ        ∫=+3.96822 μJ        ∫=+198.935 μJ
Δt=+16.0150 ms       Δt=+23.0010 ms       Δt=+90.5355 ms

μ=+7.67977 mA        μ=+0.22012 mA        μ=+2.94766 mA
σ=+2.83995 mA        σ=+8.04441 mA        σ=+8.16249 mA
min=+1.44239 mA      min=-39.2040 mA      min=-6.13298 mA
max=+39.6218 mA      max=+97.4183 mA      max=+10.2014 mA
p2p=+38.1794 mA      p2p=+136.622 mA      p2p=+16.3343 mA
∫=+122.991 μC        ∫=+5.06315 μC        ∫=+266.867 μC
Δt=+16.0150 ms       Δt=+23.0010 ms       Δt=+90.5355 ms

μ=+742.555 mV        μ=+661.237 mV        μ=+692.827 mV
σ=+3.31384 mV        σ=+46.7567 mV        σ=+53.8934 mV
min=+735.701 mV      min=+622.042 mV      min=+629.831 mV
max=+757.099 mV      max=+783.058 mV      max=+746.551 mV
p2p=+21.3986 mV      p2p=+161.016 mV      p2p=+116.719 mV
Δt=+16.0150 ms       Δt=+23.0010 ms       Δt=+90.5355 ms
```

8 Adding power optimizations in your application

The power mode handling used in this application can be adapted to any application. Where there are different application states defined, where each of them has different power settings. Each application state has a structure with the configuration registers that specify which peripherals or modules will remain powered during the low power modes.

```
typedef struct _state_power_config
{
    uint32_t pd_sleep_config[4];
} state_power_config_t;

static state_power_config_t s_PowerConfig[MAX_APP_STATE];
```

The application will then select the corresponding power configuration for each state.

```
void APP_Idle(uint32_t xModifiableIdleTime)
{
    /* Enter Deep Sleep mode if allowed (DSP is idle) */
    if (PWR_DeepSleepAllowed())
    {
        POWER_EnterDeepSleep(&s_PowerConfig[s_AppState].pd_sleep_config[0]);
    }
    else
    {
        BOARD_DisablePeripheralsDuringSleep();
        POWER_EnterSleep();
        BOARD_RestorePeripheralsAfterSleep();
    }
}
```

8.1 Important Notes to Remember

Please refer to “Designing Power-Optimized Voice UI Application on i.MX RT595” application note (AN13758) for further details.

9 Conclusion

This lab described the power optimized voice UI application and demonstrated an approach to do power optimization techniques on the i.MX RT500.