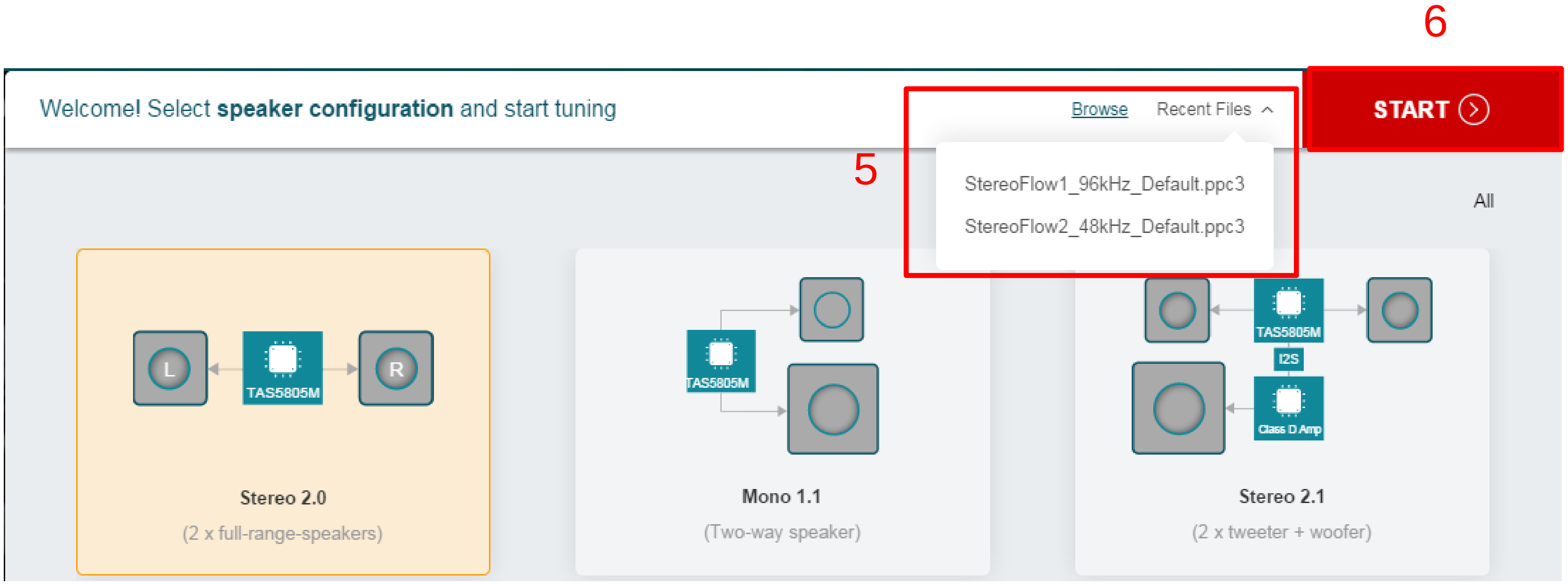
How to Generate a Header File for TAS5805M in PPC3

Step 1

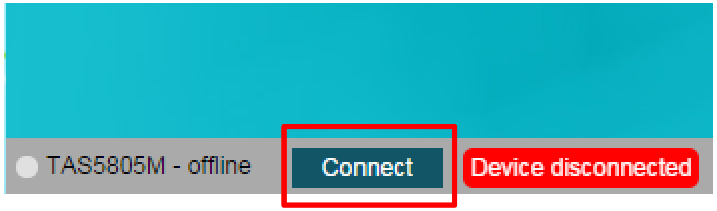
1. Connect a TAS5805MEVM to your PC.
2. Power on TAS5805MEVM.
3. Plug in a Micro USB cable from the PC to TAS5805MEVM.
4. Launch PPC3 and go to TAS5805M app.
5. Load your tuning file(.ppc3).
6. Click the START button.



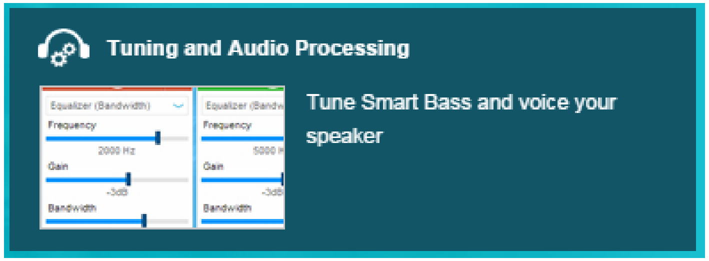
Welcome! Select **speaker configuration** and start tuning

Step 2

1. Click the Connect button on the bottom.



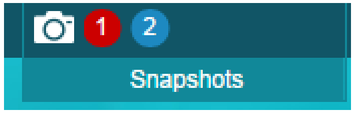
2. Open **Tuning and Audio Processing**. This will load tuning settings to the target TAS5805M device on the EVM.



Tuning and Audio Processing

Tune Smart Bass and voice your speaker

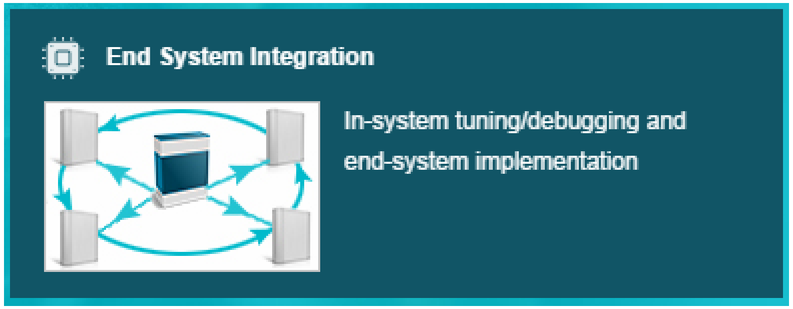
3. Select the desired **Snapshot** if any has been saved before**.**



4. Make sure your tuning settings are all good.

Step 3 (1 / 3)

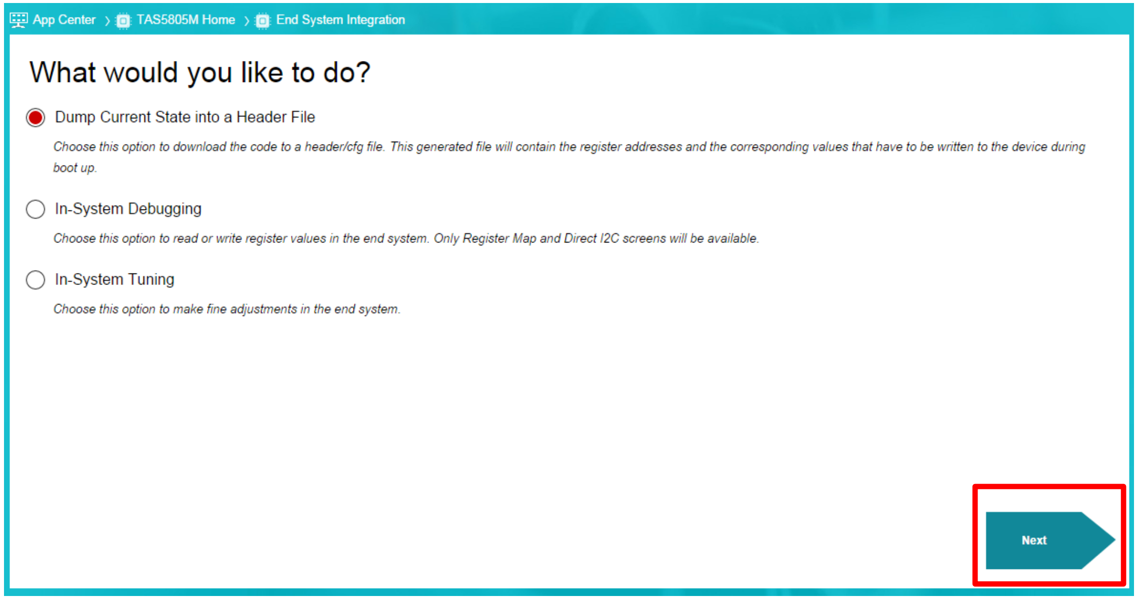
1. Go into **End System Integration**.



End System Integration

In-system tuning/debugging and end-system implementation

2. Select **Dump Current State into a Header File** and click the Next button.



App Center › TAS5805M Home › End System Integration

What would you like to do?

● Dump Current State into a Header File

Choose this option to download the code to a header/cfg file. This generated file will contain the register addresses and the corresponding values that have to be written to the device during boot up.

○ In-System Debugging

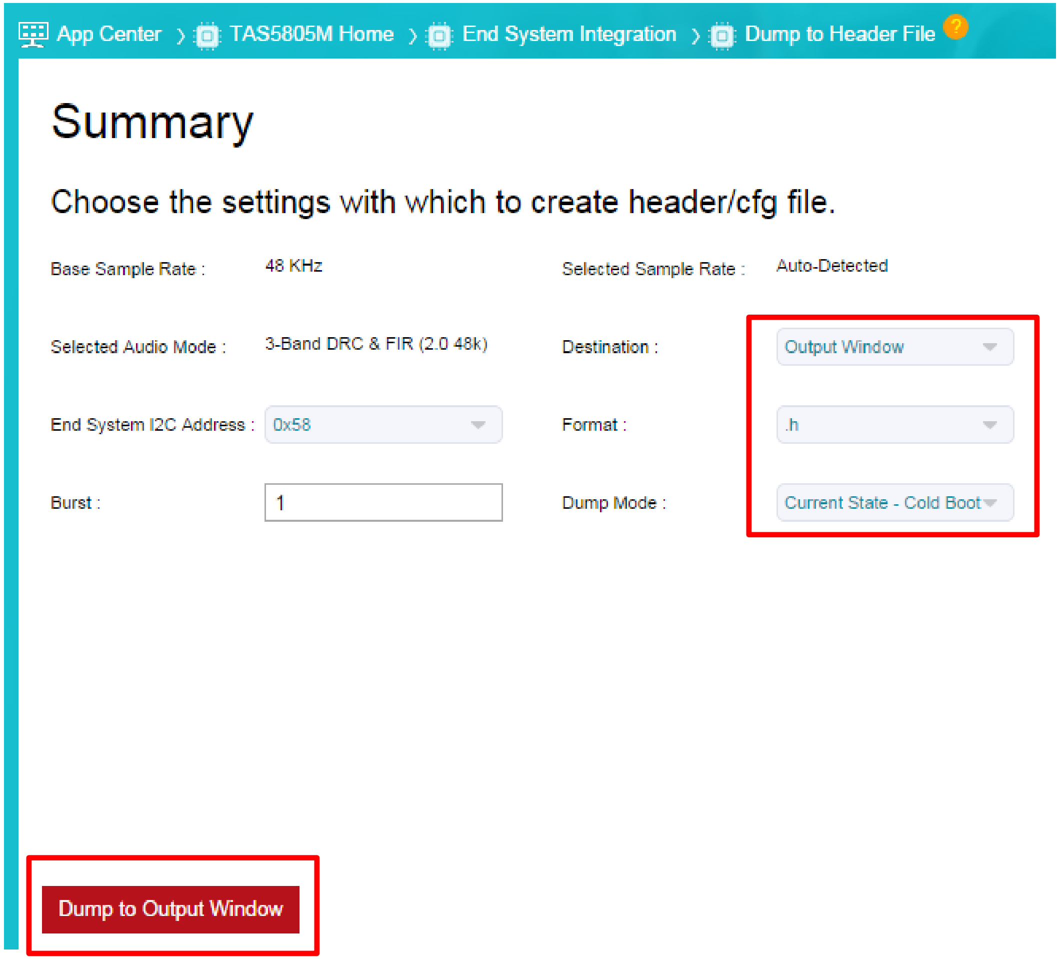
Choose this option to read or write register values in the end system. Only Register Map and Direct I2C screens will be available.

○ In-System Tuning

Choose this option to make fine adjustments in the end system.

Step 3 (2 / 3)

3. Make sure the right Destination, Format and Dump Mode are selected. Click the **Dump to Output Window** button, as shown below.



App Center › TAS5805M Home › End System Integration › Dump to Header File

Summary

Choose the settings with which to create header/cfg file.

Base Sample Rate : 48 KHz

Selected Sample Rate : Auto-Detected

Selected Audio Mode : 3-Band DRC & FIR (2.0 48k)

Destination : Output Window

End System I2C Address : 0x58

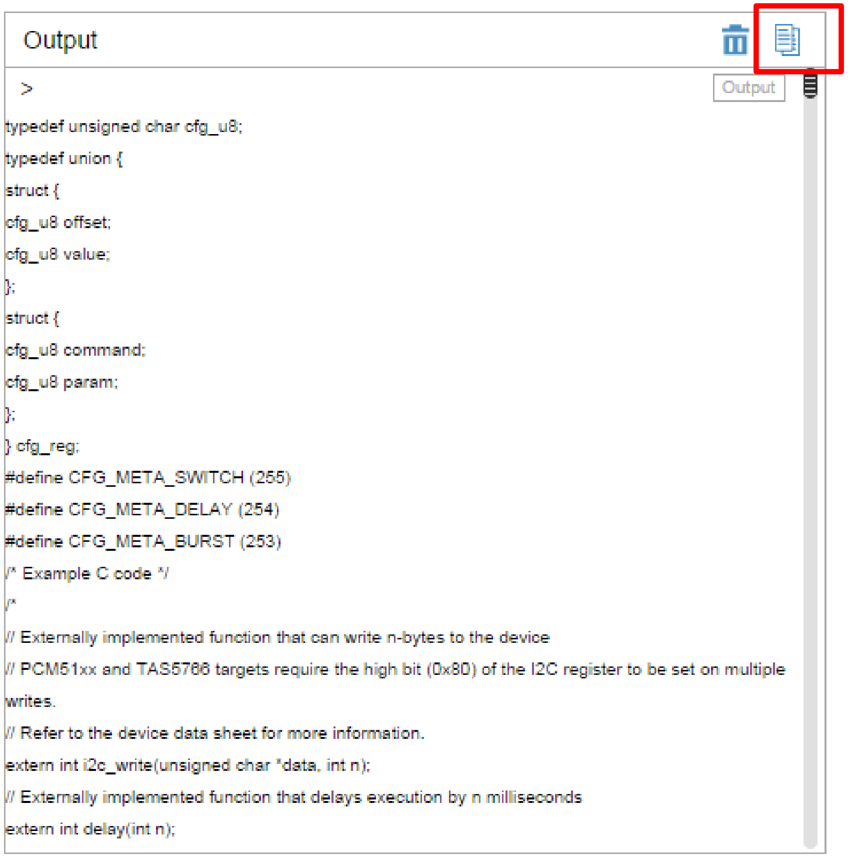
Format : .h

Burst : 1

Dump Mode : Current State - Cold Boot

Step 3 (3 / 3)

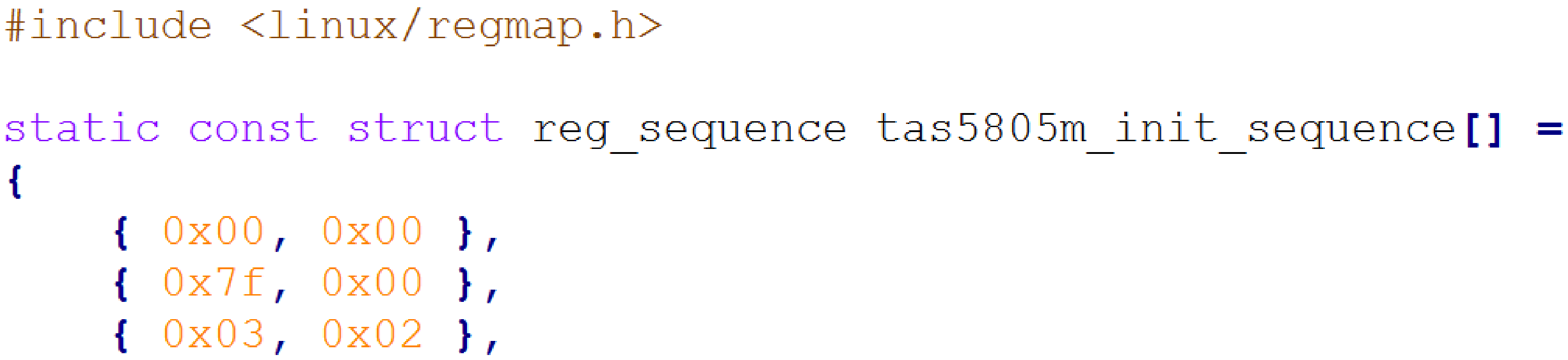
4. Click the icon on the top right corner of the Output window and paste all the contents to a blank header file.



```
>
typedef unsigned char cfg_u8;
typedef union {
struct {
cfg_u8 offset;
cfg_u8 value;
};
struct {
cfg_u8 command;
cfg_u8 param;
};
} cfg_reg;
#define CFG_META_SWITCH (255)
#define CFG_META_DELAY (254)
#define CFG_META_BURST (253)
/* Example C code */
/*
// Externally implemented function that can write n-bytes to the device
// PCM51xx and TAS5766 targets require the high bit (0x80) of the I2C register to be set on multiple
writes.
// Refer to the device data sheet for more information.
extern int i2c_write(unsigned char *data, int n);
// Externally implemented function that delays execution by n milliseconds
extern int delay(int n);
```

Step 4

1. Make a few necessary changes so that the header file can be used with the driver code. Se the example below.



```
#include <linux/regmap.h>

static const struct reg_sequence tas5805m_init_sequence[] =
{
    { 0x00, 0x00 },
    { 0x7f, 0x00 },
    { 0x03, 0x02 },
```