Generating Target Files

Audio Weaver Designer enables the user to create various files that can be deployed to or used by the target processor. This feature is accessed in Designer with:

Tools ---> Generate Target Files.

Generate Target Files gives us a popup window allowing generation of 7 types of files plus some control options. We first summarize the file types and control options, then we show how to implement this functionality with a Matlab script.

Target File Types

1.)**Audio Weaver Script, (.aws)**

- Contains all information of an AWE layout, (.awd file)
- Human readable format
- Contains valid commands which AWE Server understands and can execute
- Commands include information about the modules used, connections between them, and parameter settings

2.)**Audio Weaver Binary, (.awb)**

- Contains all information of an AWE layout, (.awd file)
- Compiled version of a .aws file
- Not human readable
- Easier for an embedded target's firmware to decode than .aws
- Most commonly used in production on embedded targets

3.)**Tuning Symbol File, (.tsf)**

- Contains the symbols needed when connecting to a running target
- Essential for using **Tools --> Attach to Running Target** in AWE Designer
- .tsf file allows Designer to connect to the layout running on the target and manipulate it as if you had loaded that layout directly via Designer

4.)**ControlInterface.h**

- Provides symbolic information which allows external code to interact with and control an AWE layout running on a target processor
- C/C++ header format with #define's for all relevant symbols

- Allows reading/writing of module variables and properties
- Works with standalone external code running on the target device, (AWE Designer/AWE Server not needed)
- 2 files are generated: _ControlInterface.h and _ControlInterface_AWE6.h. _ControlInterface_AWE6.h. is for legacy compatibility with the AWE 6 product line. The AWE 6 product line was in use for almost four years before being replaced by AWE 8.

5.)**initAWB.c,h Files**

- Produces 2 files: XXX_initAWB.c and XXX_initAWB.h
- These files contain all the information of the entire Audio Weaver Binary (.AWB) representation of your layout, structured as an initialized C array
- XXX_initAWB.c and XXX_initAWB.h can be directly integrated and compiled into targtet firmware
- Enables standalone loading of an AWE layout onto embedded processors which do not have a file system.

6.)**ModuleList.h**

- ModuleList.h is a C/C++ header file containing a list of all modules used in a layout
- Includes only modules necessary for the layout to run. Excludes unused modules and modules with Debug Status = 'Debug'
- Specifies which modules must be included in the final target firmware to support a given .awb file
- Allows a Board Support Package (BSP) author to build an AWE Core target with only the minimum set of modules required.
- Enables optimization of memory usage by excluding unnecessary modules
- In the case of multiple AWE instances a ModuleList.h file is produced for each instance detailing the modules needed for that instance

7.)**HeapSize.h**

- Audio Weaver manages its own set of 3 or 4 internal memory heaps (FAST, FASTB, SLOW, SHARED)
- HeapSize.h specifies the minimum sizes of each heap required for the AWE layout

- In the case of multiple AWE instances a HeapSize.h file is produced for each instance detailing minimum heap sizes needed for that instance
- Helps BSP developers allocate the appropriate amount of memory for the AWE Core's heaps. Helps to trim the heap sizes defined in the BSP if they are larger than needed

Target Files and Custom Object IDs

You should assign a Custom Object ID to a module in an AWE layout if you intend to control that module at run-time in your deployed product. Custom IDs will then be included in the target files:

- .aws
- .awb
- .tsf
- initAWB.c,h
- ControlInterface.h --> ONLY modules with Custom Object IDs will be included in this file. Thus for using this file, it is mandatory to assign Custom Object IDs to all modules you want to interact with on the layout

Enable Audio Checkbox

- When checked, (which is the default) this appends an audio_pump command to your generated .awb (Audio Weaver Binary) and .aws (Audio Weaver Script) files. The appended audio_pump command will initiate audio processing upon loading.
- When unchecked no audio_pump command is appended and upon loading audio processing will not begin until a manual audio_pump command is issued. This could be controlled by a specific event like a button push. The BSP must provide a mechanism to allow starting the audio pump after loading.

Split multi-layout Checkbox

- Unchecked is the default
- Designed to work with multi-instance layouts
- When checked separate .awb files will be generated for each instance in a multi-instance layout
- Max Num Parts field gives control over what instances will be in their own .awb We get a seperate .awb generated for each instance up to Max Num Parts, and any

additional instances are included in the last .awb. So if we have 4 instances, (indexed 0-3), and we set MaxNumParts to 2 then we get 2 .awb's generated. The first .awb contains instance 0 and the the 2nd .awb contains instances 1,2,3. Likewise if we have 4 instances and set Max Num Parts to 3 we will get instance 0 in the first .awb, instance 1 in the next .awb and instances 2,3 in the last .awb. This allows control over which instances can be loaded to distinct cores on an embedded target.

- Supports progresive loading

Create Lookup Table Command Checkbox

- New Feature added in Designer version 8.D.2.5
- Unchecked is the default
- When checked sets AWE_INFO.buildControl.createLookup to 1
- When checked adds a create_lookup command to the generated .aws (Audio Weaver Script) and .awb (Audio Weaver Binary) files
- Allows inclusion of a lookup table in the generated .awb and .aws files to improve the efficiency of ID lookups on the target hardware

Generating Target Files via Matlab Scripts

The Matlab Automation API function **generate_target_files()** allows us to generate all target files described above programatically from Matlab.
First lets look at the function prototype for generate_target_files()

function generate_target_files(GSYS, baseName, saveDir, sArrayName, nCoreID, calledFromGUI)

```
% Generate target files based on the GSYS props options selected
%
% Inputs:
% GSYS - Global System object for the loaded layout
% GSYS.props.generateAWSCompileWindow - generate AWS file
% GSYS.props.generateTargetInfo - generate a modulelist.h file
% GSYS.props.generateHeapUsed - generate heapSize.h file
% GSYS.props.generateAWB - generate AWB file
% GSYS.props.generateTSF - generate .tsf file
% GSYS.props.generateAwbH - generate initAWB.c,h array files
```

```
% GSYS.props.generateEval - encrypt AWB file
%
% baseName - base name for generated files
% saveDir - directory for saved files
% sArrayName - C array base name - (optional) default is baseName
% nCoreID - multi-canvas core ID - (optional) default is 0
% calledFromGUI - (optional) set to true if called from Designer GUI
% default is false
```

Notes:

1.)The _ControlInterface.h files are always generated even if all the above GSYS.props fields are set to zero.

2.)generate_target_files() allows encryption of the .awb file by setting GSYS.props.generateEval = 1

Below is Matlab code to illustrate the use of this function

```
% Name of Audio Weaver layout to be controlled
fNameAWD = 'TargetFilesTest.awd';

% Set values for all function arguments
baseName = 'TT';
saveDir =
'C:\Users\rhutc\OneDrive\Desktop\DSPC\AWE_Doc_2025\MatlabAutomationAPI\TargetFil
es';
sArrayName = 'TTC';
nCoreID = 0;
calledFromGUI = 0;

% Get the GSYS structure for the AWE layout
GSYS = load_awd(fNameAWD);

% Declare global AWE_INFO after gettting GSYS
global AWE_INFO;

% Set GSYS.props fields to select files to be generated
GSYS.props.generateAWSCompileWindow = 1;    % generate .aws file
GSYS.props.generateTargetInfo = 0;    % generate a modulelist.h file
GSYS.props.generateHeapUsed = 0;    % generate heapSize.h file
GSYS.props.generateAWB = 0;    % generate .awb file
GSYS.props.generateTSF = 0;    % generate .tsf file
```

```
GSYS.props.generateAwbH = 0;     % generate initAWB.c, h array files
GSYS.props.generateEval = 0;     % encrypt .awb file

% Set values for control check boxes
%
% Enable Audio Check Box
GSYS.props.enableAudioPump = 1;
%
% Split multi-layout Checkbox
GSYS.props.generateNumParts = 4;
GSYS.props.generateSplitAWB = 0;
%
% Create Lookup Table Command Checkbox
AWE_INFO.buildControl.createLookup = 0;

% Function call to generate the files
generate_target_files(GSYS, baseName, saveDir, sArrayName, nCoreID, calledFromGUI);
```