

# AquesTalk ESP32 マニュアル

株式会社 アクエスト  
www.a-quest.com

## 概要

本文書は、音声合成ライブラリ AquesTalk ESP32 をアプリケーションに組み込んで使用するための関数定義、プログラミングの方法、注意点を示したものです。

AquesTalk ESP32 は、AquesTalk pico と AqKanji2Roman を Espressif Systems 社製のマイコン ESP32 用に移植したライブラリです。

AquesTalk pico は、ローマ字表記の音声記号列から音声波形データを生成するライブラリです。

AqKanji2Roman は、漢字仮名混じりのテキストをローマ字表記の音声記号列に変換するライブラリです。

本ライブラリを使用するには、ライセンスキーの設定が必要です。このライセンスキーを設定しない場合は、評価版として動作し、以下の制限があります。

評価版の制限

「ナ行、マ行」を指定すると、すべて「ヌ」と発声します

ライセンスの種類や購入方法は、弊社サイトのライセンスのページを参照してください。

ArduinoIDE や ESP-IDF 環境での具体的な使用方法をブログでも紹介していますので、そちらも参照ください。 <http://blog-yama.a-quest.com/>

## 仕様

ライブラリ形式	ESP32 用 static ライブラリ
動作確認環境	Arduino core for ESP32、ESP-IDF
入力データ形式	漢字仮名交じりテキスト(UTF8/UTF16LE)、または、ローマ字表記音声記号列(ASCII)
出力データ形式	WAV フォーマット(8KHz サンプリング, 16bitPCM, モノラル) データ *24KHzサンプリングにアップサンプリング機能を含む
声種	picoF4 Rev.2
関数 I/F	C 関数呼び出し
必要メモリサイズ	ROM: 約 200KB, RAM: 約 21KB (漢字仮名交じり文からの合成の場合 辞書データ別途) ROM: 約 28KB, RAM: 約 500B (音声記号列からの合成のみの場合)

言語処理部

CAqK2R\_Create aquestalk.h

説明	言語処理部の初期化
構文	int <b>CAqK2R_Create</b> (uint8_t *workbuf, int sizeWorkBuf)
引数	
<i>workbuf</i>	ワークバッファ。呼び出し側で確保。サイズは SIZE_AQK2R_MIN_WORK_BUF 以上
<i>sizeWorkBuf</i>	workbuf のサイズ[byte]
戻り値	0:正常終了 それ以外:エラーコード参照

CAqK2R\_Release aquestalk.h

説明	言語処理部の解放
構文	void <b>CAqK2R_Release</b> ()
引数	なし
戻り値	なし

CAqK2R\_Convert aquestalk.h

説明	漢字かな混じり文(UTF8)をローマ字音声記号列に変換
構文	int <b>CAqK2R_Convert</b> (const char *kanji_utf8, char *roman, int nBufRoman)
引数	
<i>kanji_utf8</i>	漢字かな混じり文テキスト(UTF8,BOM 無)
<i>roman</i>	ローマ字音声記号列が返る文字列配列
<i>nBufRoman</i>	roman の配列のサイズ
戻り値	0:正常終了 それ以外:エラーコード参照

## CAqK2R\_ConvertW

aquestalk.h

説明	漢字かな混じり文(UTF16LE)をローマ字音声記号列に変換
構文	int <b>CAqK2R_ConvertW</b> (const char16_t * <i>kanji</i> , char * <i>roman</i> , int <i>nBufRoman</i> )
引数	
<i>kanji</i>	漢字かな混じり文テキスト(UTF16LE)
<i>roman</i>	ローマ字音声記号列が返る文字列配列
<i>nBufRoman</i>	<i>roman</i> の配列のサイズ
戻り値	0:正常終了 それ以外:エラーコード参照

## 規則合成部

## CAqTkPicoF\_Init

aquestalk.h

説明	初期化
構文	uint8_t <b>CAqTkPicoF_Init</b> (uint32_t * <i>buf</i> , uint16_t <i>lenFrame</i> , const char * <i>key</i> )
引数	
<i>buf</i>	ワークバッファ。呼び出し側で確保。サイズは AQ_SIZE_WORKBUF
<i>lenFrame</i>	SyntheFrame()の呼び出し毎に生成するサンプル数を指定(30-320)
<i>key</i>	ライセンスキー文字列を指定。NULL 指定で評価版としての動作(制限有)。
戻り値	0:正常終了 それ以外:エラーコード参照

## CAqTkPicoF\_SetKoe

aquestalk.h

説明	音声記号列をセット。 この関数呼び出し後、SyntheFrame()を連続して呼び出して音声波形を生成する
構文	uint8_t <b>CAqTkPicoF_SetKoe</b> (const uint8_t * <i>koe</i> , uint16_t <i>speed</i> , uint16_t <i>lenPause</i> )
引数	
<i>koe</i>	ローマ字音声記号列(NULL 終端)。呼び出し後、解放可能。

<i>speed</i>	発話速度 [%] 50-300 の間で指定    slow:50 fast:300 default:100
<i>lenPause</i>	最後のポーズ長[sample]。0xffffU を指定すると、内部の標準ポーズ長。(デフォルト)。
戻り値	0:正常終了    それ以外:エラーコード参照

CAqTkPicoF\_SyntheFrame

aquestalk.h

説明	音声波形を生成
構文	uint8_t <b>CAqTkPicoF_SyntheFrame</b> (short *wav, uint16_t *pLen)
引数	
<i>wav</i>	音声波形出力バッファ(lenFrame×2byte 以上)。 最終フレームは *pSize<lenFrame の場合であっても残りのデータはゼロパディングされる
<i>pLen</i>	生成したデータのサンプル数を返す。 通常、最終フレーム以外は lenFrame で指定した値。
戻り値	0:正常終了    1:EOD    それ以外:エラーコード参照

リサンプリング部

AqResample\_Reset

aquestalk.h

説明	内部変数の初期化。 音声合成の単位ごとに呼び出す。
構文	void <b>AqResample_Reset</b> ()
引数	なし
戻り値	なし

AqResample\_Conv

aquestalk.h

説明	アップサンプリング処理(3 倍アップサンプリング) 1 サンプルずつ与えると、3 サンプルずつ返す
構文	void <b>AqResample_Conv</b> (short <i>sin</i> , short * <i>sout</i> )
引数	

sin	音声波形の 1 サンプルを指定
sout	音声波形の 3 サンプル返す。short[3]の配列を呼び出し側で用意。
戻り値	なし

## アプリ側で用意する関数

言語処理部の関数を使用する場合は、以下の関数を別途実装する必要があります。これらの関数には辞書データ(aqdic\_m.bin)にアクセスする機能を実装します。これらの関数は言語処理部から呼び出されます。

C でなく C++で実装する場合は、関数名がマングルされないように、extern "C" を指定してください。

### aqdic\_open

説明	辞書データアクセスの初期化 CAqK2R_Create()内から一度だけ呼び出される
構文	extern "C" size_t <b>aqdic_open()</b>
引数	なし
戻り値	仮想的な辞書データの先頭アドレスを返す(0 以外の任意の値。4byte アライメント) 初期化に失敗したときは 0 を返す

### aqdic\_close

説明	辞書データアクセスの終了 CAqK2R_Release()内から一度だけ呼び出される
構文	extern "C" void <b>aqdic_close()</b>
引数	なし
戻り値	なし

### aqdic\_read

説明	辞書データの読み込み CAqK2R_Convert()/CAqK2R_ConvertW()から複数回呼び出される
構文	extern "C" size_t <b>aqdic_read</b> (size_t pos, size_t size, void *buf)
引数	

<i>pos</i>	読み込むデータの先頭アドレス[byte] = aqdic_open()で返したアドレス + 辞書データのオフセット
<i>size</i>	読み込むデータのサイズ[byte]
<i>buf</i>	読み込むデータ配列 uint8_t(size)
戻り値	読み込んだバイト数

---

以下に、辞書データを SD カード上のファイルとして配置し、そこから読み込む実装例を示します。

aqdic\_access.cpp

```

1  #include <SD.h>
2  const uint8_t SD_CS = 5;
3  const size_t ADDR_ORG = 0x10001000;
4  #define FILE_DIC "/aq_dic/aqdic_m.bin"
5
6  static File fp;
7
8  extern "C" size_t aqdic_open()
9  {
10     if(!SD.begin(SD_CS)) return 0; // err
11
12     fp = SD.open(FILE_DIC);
13     if(!fp) return 0; // err
14     return ADDR_ORG; // ok
15 }
16
17 extern "C" void aqdic_close()
18 {
19     if(fp) fp.close();
20 }
21
22 extern "C" size_t aqdic_read(size_t pos, size_t size, void *buf)
23 {
24     fp.seek(pos-ADDR_ORG);
25     return fp.read((uint8_t*)buf,size);
26 }
```

2 行目：SD カードの CS のピンアサイン

3 行目：仮想的な辞書データの先頭アドレスを指定（NULL 以外の任意。但し 4byte アライメント）

4 行目：辞書データファイルの SD メモリカード上の配置場所（ここでは、aq\_dic フォルダの下）

## 規則合成部の関数呼び出しシーケンス

次コードを例に関数の呼び出しシーケンスを示します(コードはシーケンスを示すためのもので、実際に有効なソースコードではありません)。

1. CAqTkPicoF\_Init()にて初期化とワークバッファを指定。
2. CAqTkPicoF\_SetKoe()にて合成するメッセージを指定。
3. CAqTkPicoF\_SyntheFrame()を繰り返し呼び出し、生成した 1 フレーム分の波形データを D/A に出力する。
4. 3.を、戻り値 1 (End of Data)が返るまで繰り返す。

さらに他のメッセージを合成する場合は、2.のステップから繰り返します。ただし、ワークバッファを解放した場合は 1.のステップから行います。

なお、音声合成を他の処理と並列に行う場合は、4.のループ処理を展開するようにします。

```
1  #define  FRAME_LEN 32
2  uint32_t buf[AQ_SIZE_WORKBUF];
3  const uint8_t koe[]="konnichiwa.";
4
5  uint8_t iret;
6  CAqTkPicoF_Init(buf, FRAME_LEN, "xxx-xxx-xxx");
7  iret = CAqTkPicoF_SetKoe(koe, 100, 256);
8  if(iret!=0) <<<error proc>>>
9
10 for(;;){
11     short wavFrame[FRAME_LEN];
12     uint16_t size;
13     iret = CAqTkPicoF_SyntheFrame(wavFrame, &size);
14     if(iret==1) break; // End of Data
15     if(iret!=0) <<<error proc>>>
16     <<< Copy "wavFrame" data to D/A output buffer. Length is "size" or FRAME_LEN >>>
17 }
```

ArduinoIDE や ESP-IDF 環境での具体的な使用法は、ブログで公開していますので、「AquesTalk ESP23」などのキーワードで検索して、それを参照ください。

## 音声記号列からの音声合成

AquesTalk ESP32 は、漢字仮名交じりのテキスト情報から音声合成が可能ですが、音声記号列から音声を合成するだけで足りる場合は、言語処理部の関数を使用しないことで、大幅に ROM/RAM サイズを小さくできます。

音声記号列の詳細は、当社サイトのダウンロードのページの「ローマ字音声記号列仕様書」を参照してください。

なお、漢字仮名混じり文から音声記号列に変換する AqKanji2Roman ライブラリは、各種プラットフォーム版が用意されていますので、音声記号列への変換はサーバーで行い、これを転送し、端末側で音声出力するといったシステム構成も可能です。

## エラーコード表

言語処理部の関数が返すエラーコードの内容は、次の通りです。

値	内容
100	その他のエラー
101	関数呼び出し時の引数指定エラー
102	ワークバッファ不足(入力テキストが長すぎる)
103	出力の音声記号列バッファ不足
104	CAqK2R_Create()で初期化されていない
200	辞書データのオープンエラー
210-218	辞書データが正しくない(フォーマットエラー)
220-222	辞書データが正しくない(バージョンエラー)

規則合成部の関数が返すエラーコードの内容は、次の通りです。

値	内容
100	その他のエラー
105	音声記号列に未定義の読み記号が指定された
106	音声記号列のタグの指定が正しくない
107	タグの長さが制限を越えている(または[>]が見つからない)
121	CAqTkPicoF_Init()で初期化されていない
122	関数の引数の指定が正しくない
123	ワークバッファが外部から書き換えられた
200	音声記号列が長すぎる

## 文書履歴

日付	版	変更箇所	更新内容	更新者
2018/03/22	1.0		新規作成	N.Y
2018/08/08	2.0		言語処理部追加	N.Y