iWASM Source Code User Guide

Intel Micro Runtime Team, SSG

Contents

[1 Integration process 1](#_Toc534811416)

[2 Platform layer adaption 1](#_Toc534811417)

[3 WASM Application APIs 1](#_Toc534811418)

[3.1 iWASM built-in application API 2](#_Toc534811419)

[3.2 Platform APIs 2](#_Toc534811420)

[4 Develop a WASM Application 4](#_Toc534811421)

# Integration process

iWASM is a WebAssembly VM designed for embedded environment, so it took the portability and customization as the a key design goal.

There are two necessary steps for integrating the iWASM for a target board:

1. Add or modify the iWASM platform layer for the board OS, and complete the full image build with iWASM integrated
2. Export the application APIs for WASM application development

Notes: This document is mostly about C based WASM application. It should be no technical problem for supporting other programming languages such as RUST.

# Platform layer adaption

To be finished.

# WASM Application APIs

There are 3 sources of APIs for programming the WASM application:

1. **iWASM built-in API**: iWASM has already provided a minimal API set for applications. The minimal API declaration header file is “lib/lib-base.h”.
2. **Application ported API**: The application can include any C source code from any 3rd party library, and use the emcc to compile it and integrate the implementation of referred APIs into the application package.
3. **Platform APIs**: The board vendor can choose what platform APIs to be exposed to WASM applications during making the board firmware.

## iWASM built-in application API

The built-in APIs are defined in “lib/lib-base.h”, WASM application project should include this header file. The API set is listed as below:

|  |
| --- |
| void \*malloc(size\_t size);  void \*calloc(size\_t n, size\_t size);  void free(void \*ptr);  int memcmp(const void \*s1, const void \*s2, size\_t n);  void \*memcpy(void \*dest, const void \*src, size\_t n);  void \*memmove(void \*dest, const void \*src, size\_t n);  void \*memset(void \*s, int c, size\_t n);  int putchar(int c);  int snprintf(char \*str, size\_t size, const char \*format, ...);  int sprintf(char \*str, const char \*format, ...);  char \*strchr(const char \*s, int c);  int strcmp(const char \*s1, const char \*s2);  char \*strcpy(char \*dest, const char \*src);  size\_t strlen(const char \*s);  int strncmp(const char \* str1, const char \* str2, size\_t n);  char \*strncpy(char \*dest, const char \*src, unsigned long n); |

## Platform APIs

This section explains how to export “Platform API” for developing WASM application in the target platform. “Platform API” can be any function defined by the OS or the board firmware code.

iWASM implemented a framework for developers to export APIs. The procedure to expose the platform APIs:

1. Create a header file to declare the APIs for WASM application source project to include.

2. Modify “lib/lib-export-template.c” from the iWASM repo, and add the exported function names into the array “extended\_native\_symbol\_defs”.

The pre-defined two MACROs below should be used to declare a function export:

|  |
| --- |
| #define EXPORT\_WASM\_API(symbol) {#symbol, symbol}  #define EXPORT\_WASM\_API2(symbol) {#symbol, symbol\_##wrapper} |

The type of array “extended\_native\_symbol\_defs[]” is defined as below:

|  |
| --- |
| typedef struct NativeSymbol {  const char \*symbol;  void \*func\_ptr;  } NativeSymbol; |

**NOTE:** please **DO NOT** edit any other files except for “lib/lib-export-template.c” under iWASM repo.

Example for Zephyr OS:

a. lib/lib-export-impl.c

|  |
| --- |
| #include <zephyr.h>  #include <kernel.h>  #include <gpio.h>  static void customized()  {  // your code  }  static int  gpio\_pin\_configure\_wrapper(struct device \*port, u32\_t pin, int flags)  {  return gpio\_pin\_configure(port, pin, flags); // a Zephyr OS API  } |

b. lib/lib-export-dec.h

|  |
| --- |
| #ifndef \_LIB\_EXPORT\_DEC\_H\_  #define \_LIB\_EXPORT\_DEC\_H\_  #ifdef \_\_cplusplus  extern "C" {  #endif  void customized();  int gpio\_pin\_configure\_wrapper(struct device \*port, u32\_t pin, int flags);  #ifdef \_\_cplusplus  }  #endif  #endif |

c. lib/lib-export-template.c

|  |
| --- |
| #include "lib-export-template.c"  #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #include "lib-export.h"  NativeSymbol extended\_native\_symbol\_defs[] = {  EXPORT\_WASM\_API(customized),  EXPORT\_WASM\_API2(gpio\_pin\_configure)  }; |

# Develop a WASM Application

Application can include the iWASM built-in APIs header file and platform provided header files in the source project, then build the WASM package by using the emcc compiler.

An WASM application example for Zephyr OS:

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <stdlib.h>  #include <stdlib.h>  #include “lib-base.h” // provided by us  #include “lib-export-dec.h” // provided by platform developers  int main(int argc, char \*\*argv)  {  int I;  char \*buf = “abcd”;  i = strlen(buf); // common API provided by us  customized(); // customized API provided by platform developers  return i;  } |