Classic 8080

Edit New page

Jump to bottom

Phillip Stevens edited this page on Feb 14 · 28 revisions

After a few decades of not supporting 8080 code generation, z88dk can once again generate code that will run on 8080 and 8085 based computers.

The z88dk classic library also supports compiler intrinsic (`l_`) functions optimised for 8080, 8085, gbz80, or z80 CPUs. Specifically, the undocumented 8085 instruction extensions to the 8080 instruction set are supported and are used where appropriate, and gbz80 specific instructions are also issued where appropriate.

Supported targets

8080 CPU targets

- PMD85
- Vector06c
- Corvette
- Mikro 80
- Specialist
- DAI
- Radio 86
- Altair 8800
- LVIV
- Krokha
- Sol20

Additionally, the CP/M target can generate 8080 binaries. To do so, add the option `-clib=8080` to the command line. For example:

```
zcc +cpm -clib=8080 adv_a.c
```

A .bin file will be generated that should run on 8080 based CP/M machines.

8085 CPU Targets

- TRS80 Model 100
- Micro85
- RC2014 using the 8085 CPU Module

The RC2014-8085 support consists of two subtypes specifically for use with the 8085 CPU Module.

The `basic85` subtype produces a HEX file for use with the Microsoft Basic ROM (the appropriate floating point library can be linked for use either with or without the APU Module), and can be uploaded to the RC2014 using the `HLOAD` Basic keyword.

Additionally, both the RC2014 `basic` and `basic85` subtypes are supported by the `z88dk-ticks` emulator. Where the resulting binary just needs to be titled `rc2014.bin` , and emulated with the correct machine type.

```
zcc +rc2014 -subtype=basic85 my_c_file.c my_asm_file.asm -o my_hex_file -create-app
```

The `acia85` subtype produces a HEX ROM file that provides serial interfaces using the RC2014 ACIA Serial Module.

```
zcc +rc2014 -subtype=acia85 my_c_file.c my_asm_file.asm -o my_hex_rom_file -create-app
```

Limitations

The 8080/5 CPU implements a subset of the z80 instruction set that the compilers and libraries within z88dk have traditionally targeted. As a result not all features are available. If you use a feature that is not available then your program will fail to link.

`<stdio.h>`

The following features from `<stdio.h>` are not available in the 8080/5 library:

- `funopen()` and stdio pluggable device support
- Floating point support for the `scanf` family

`<stdlib.h>`

The following features from `<stdlib.h>` are not available in the 8080/5 library:

- z88dk extension: `extract_bits`

`<math.h>`

There are two maths libraries that are available for the 8080/8085: `dai32` , extracted from the DAI ROM, and `mbf32` Microsoft Basic Floating Point, extracted from Microsoft Basic 4.7 and optimised for the 8080, 8085, gbz80, and z80 CPUs.

The RC2014 8085 has access to the `am9511_8085` maths library when it is equipped with an APU Module. This maths library supports `long` and `float` operations with the APU, and provides a substantial performance gain over software floating point support.

Aliases are available for specific CPU types to simplify the use of any of these maths libraries.

z88dk extension libraries

The following extension libraries are not available in the 8080/5 library:

- Allocation: balloc
- Algorithm/adt: All
- Debug library

Compiler support

Only `sccz80` supports generation of 8080 and 8085 code.

- i8080/5 Support
- Homebrew hardware quickstart
- Retargetting
- Building the libraries
- Pragmas
- Adding to Classic

New Library

- Introduction
- Library Configuration
  - clib_cfg.asm
  - clib_target_cfg.asm
  - Rebuilding the Library
- CRT
  - crt configuration
  - pragma overrides
  - memory map
  - user initialization and exit code
  - User-Supplied Crt
- Header Files
- Assembly Language
- Library in Depth
- Embedded Platform
- Adding to NewLib

Misc

- Benchmarks
- Datatypes
- Debugging
- Decompression
- More than 64k
- Deficiencies
- Compiling Larger Applications
- Importing routines written in 8080 assembly mnemonics
- Using CP/M libraries in REL format with z88dk
  - Linking external libraries
  - Linking Basic, Fortran, Pascal Programs with z88dk
- Writing optimal code
- Speeding up Compilation

Mixing C and Z80 Assembler

- Calling Conventions
- The Stack frame (Parameter passing)
- Sharing Code by Creating Libraries
- Inline Assembler

Tools

- The Compiler Frontend (*zcc*)
- The Assembler / Linker (*z80asm*)
- Object and Library file Dumper (*z80nm*)
- Object and Library file Manipulator (*zobjcopy*)
- *copt*
- *ticks* emulator
- z88dk-gdb debugger
- Disassembler
- Various tools

Clone this wiki locally

```
https://github.com/z88dk/z88dk.wiki.git
```