How to change instruction set from x86 to Z80 in FASM

This small manual was made to integrate the Z80 instruction set in newer (or maybe older for some reason) versions of FASM. There is a complete version with release 1.69.35 available at the forum / board of flatassembler (FASM) here:

http://board.flatassembler.net/forum.php?f=22 (Non-x86-architecture)

The following files are needed or have to be changed:

Z80.INC
This file has to be used instead of X86_64.INC and contains the main code for writing Z80 binary code regarding the Zilog syntax and registers. A manual of Zilog can be found here:

http://www.zilog.com/docs/z80/ps0178.pdf

This is a version with scanned datasheet from Z80 cpu. There is PDF with name UM0080 existing on the Zilog Website in better quality (new written) but contains many errors in the displayed binary code for instructions.

TABLESZ80.INC
This file has to be used instead of TABLES.INC and contains the symbols, register names and instruction names regarding Zilog syntax. The file Z80.INC contains the code for interpreting the symbols and converts them into binary code.

Z80.INC and TABLESZ80.INC belong together and have to replace the existing files and should be copied in the SOURCE directory.These files are included in the source code depending to the version you are using. So the include lines in have to be changed in

SOURCE\IDE\FASMW\FASM.INC => for WIN IDE version
SOURCE\IDE\FASMD\FASMD.ASM => for DOS IDE version
SOURCE\IDE\DOS\FASM.ASM => for DOS command line version
SOURCE\IDE\LIBC\FASM.ASM => for C library (fasm.o)
SOURCE\IDE\LINUX\FASM.ASM => for LINUX
SOURCE\IDE\WIN32\FASM.ASM => for WIN command line version

Be sure to replace the includes at the correct lines because DOS versions are devided into segments.

But two more modifications are needed:

PARSER.INC
The parser is reading the TABLES file and because Zilog has symbols with only one character (register names and flag names) the following section has to be changed for support of one-char-symbols (x86 version supports only symbols with minimum 2 chars):

```
get_symbol:
      push  esi
      mov   ebp,ecx
      call  lower_case
      mov   ecx,ebp
      cmp   cl,11
      ja    no_symbol
;
; 14.01.2011
; changed by Karl-Heinz Dahlmann
; support of ZILOG syntax for symbols with only one character (registers
and flags)
;
      sub   cl,1
;     sub   cl,2 (original line)
;
; end of change
;
```

Simply look for get_symbol: function in PARSER.INC.

PREPROCE.INC

This is the pre-processor of FASM. Due to a different syntax from Zilog for memory addressing in comparison to Intel there is a converting function needed which replace () with [].

In Intel syntax memory addressing is marked like mov ax,[variable] – in Zilog syntax a memory addressing is written like LD A,(variable). The modification changes () to [] only when compiling source code. The function will not change the source code when autosave source file (it is done after saving source). It is possible to use [] for memory addressing even with Z80 instruction set but it is not recommended to use because source file would not be compatible to other cross assemblers.

Change of () to [] is avoided for complex calculations of immediates with using () but only when () are not obsolete. The following example will be unchanged:
LD A,(variable1+variable2)*varsize – or – LD A,varsize*(variable1+variable2)

The following example is changed to memory addressing instead of immediate addressing:
LA A,(variable1*varsize)
But in this case () are obsolete. If somebody wants he can write a more complex version of the function convert_syntax.

1. Implement the function convert_syntax by following include line at end of file PREPROCE.INC:

Include 'preprz80.inc'

This file contains the convert function convert_syntax.

2. Implement function call convert_syntax in function preprocess_file:

```
preprocess_file:
      push  [memory_end]
      push  esi
      mov   al,2
      xor   edx,edx
      call  lseek
      push  eax
      xor   al,al
      xor   edx,edx
      call  lseek
      pop   ecx
      mov   edx,[memory_end]
      dec   edx
      mov   byte [edx],1Ah
      sub   edx,ecx
      jc    out_of_memory
      mov   esi,edx
      cmp   edx,edi
      jbe   out_of_memory
      mov   [memory_end],edx
      call  read
      call  close
;
; 14.01.2011
; changed by Karl-Heinz Dahlmann
; support of ZILOG syntax () for memory access instead of []
;
      call  convert_syntax
;
; end of change
;
```

You can find all files in the source distribution of the last Z80 version of FASM.

Regards,

Karl-Heinz Dahlmann
81827 Munich