Zilog asz280 Assembler
(for Z80/Z180/Z280)

Overview:

The *asz280* assembler was developed within the assembler suite for small systems by Alan R. Baldwin, ASxxxx version 5.20. The assembler was derived from his Z80 version, *asz80*, but is table-driven, and shares little code anymore with that assembler. It uses the code common to all of his assemblers (asxsrc) without modification. This assembler is freely distributed under the GNU Public License, version 3, available at:

http://www.gnu.org/licenses/gpl.html

Instruction Sets:

A number of assembler directives control which instructions may be used within a section of code. One may freely switch between instruction sets by multiple uses of the following directives.

.z80	Default; only instructions for the Z80 may be used
.z180	The instruction set is expanded to include both Z80 and Z180 instructions.
.z280	z280 instructions and addressing modes may be used. Note that only User Mode instructions are in this set; privileged z280 instructions may not be used. Certain z80 instructions (such as IM *n*) are privileged on the z280 and may not be used.

Three specialized instruction sets may be enabled; viz.,

.z280p	z280 Privileged Instructions may be assembled. This also enables the z80 instructions that are privileged on the z280. This is the broadest category of instructions, such as would be used for supervisor programming.
.z280n	No I/O. Since user programs on the z280 may be restricted from using I/O instructions, this assembly mode prevents their use. This includes placing z80 I/O instructions off-limits, too.
.z80u	Enables z80 undocumented instructions. The set of instructions enabled by this directive may not be complete as of the first release of the assembler.

Addressing Mode Syntax:

DA – direct address (all)

Direct addresses are enclosed in parentheses, and may include expressions; e.g.,

```
data:     .dw       0x25

          ld   a,(data)
```

Exceptions:

```
          call subr
          jp   label
          djnz label2
```

SX – short index (all)

```
          ld   a,50(ix)       ; SDCC compatible
          ld   a,(iy-25)      ; asz280 syntax
```

Familiar to z80 programmers, indexing is off of the IX or IY register, and indices may be in the range [-128..+127]. The first form may be an expression, but it must not begin with a left paren. If parentheses are needed, use:

```
          ld   a,0+(expression)(ix)
```

The second form requires that the index register be the first term after the opening left paren.

LX – long index (z280) (‘X’ in the 280 manual)

```
          ld   a,5000(ix)
          ld   a,(iy-0x480)
          ld   a,4000(hl)
          ld   a,(hl+01000)   ;octal offset
```

Many, but not all, z280 instructions allow indexing using the HL register. LX addresses are distinguished by the magnitude of the offset. If the HL register is used, the magnitude of the offset is not significant. Likewise, if the second form is used, the index register must be the first term after the opening left paren.

SR – stack relative (z280)

```
          add  a,(sp+0x224)
          ld   a,0x224(sp)
          sbc  a,(sp)              ;offset is 0000
```

All indices are taken as signed word offsets.

RA – relative addresses (z280)

Relative addresses are used on the Z80 for certain instructions:

```
djnz label
jr   [cond,] label
```

The label offset is computed from the location counter *after* the instruction.

Relative addresses on the z280 may be used almost anywhere DA, LX, or SR addressing is permitted. Examples:

```
ld   a,[data+20]
jp   [label]
call [subr]
```

Relative addresses are distinguished by the use of square brackets, instead of parentheses. All offsets are calcuted from the beginning of the instruction following the one using the relative address. Relative address usage with the JP instruction is useful for extending the range of a JR to a long offset. Example,

```
jp   nc,label  ;direct address, require relocation
jr   nc,label  ;relative always, no relocation

jp   nc,[label]     ;specifically relative, longer
                    ; range than JR
```

Relative addressing may be use with external code references, such as:

```
.globl    ext_subr

call [ext_subr]     ;offset filled in by aslink
```

References:

1. ASxxxx Cross Assemblers – Alan R. Baldwin
2. Preliminary Technical Manual, Z280 MPU Microprocessor Unit, Zilog, 1989.