A. **Appendix for asz8 Frankenstein Assembler**

A.1 **Pseudo Operations**

A.1.1 ***Standard_Pseudo_Operation_Mnemonics***

End	END
File Inclusion	INCL INCLUDE
If	IF
Else	ELSE
End If	ENDI
Equate	EQU
Set	SET
Org	ORG
Reserve Memory	RESERVE RMB
Define Byte Data	BYTE DB FCB
Define Word Data	DW FDB WORD
Define String Data	FCC STRING
Define Character Set Translation	CHARSET
Define Character Value	CHARDEF CHD
Use Character Translation	CHARUSE

A.1.2 ***Machine_Dependent_Pseudo_Operations***

A.1.2.1 ***Register_Symbol_Definitions***

```
Label REG RegisterSymbol
Label RREG RegisterSymbol
Label REG expression
Label RREG expression
Label REG
Label RREG
```

Symbols in the register address space are defined with the REG, and RREG pseudooperations. The symbol in the label field is added to the symbol table as a register symbol. The symbols defined with the REG are single byte registers. The symbols defined with RREG are double registers, and must be on even boundries.

The operand can be in three forms.

When the operand is a register symbol, the label symbol is defined with the same register address as the symbol, with the size determined by which pseudoop is used. Aliases to working registers must use this form.

When the operand is an expression, the label symbol is defined at the value of the expression, with the size determined by which pseudoop is used. The value must be in the range 0 to 127 (or 223 for the UPC register set), or 240

to 255, and must be defined at the first pass.

With no operand, the label symbol is defined at the register address following the previous REG or RREG instruction. If a previous Register Definition used a working register as its operand, the current definition will be in the working set, otherwise it will be in the register file.

Register Symbols must be surrounded by parenthesis when used in expressions.

A.1.2.2 ***Register_Set_Selection***

CPU string

The register set can be specified in the source file with the CPU pseudooperation. The string, delimited by quotes or apostrophes, is scanned for a substring which selects which register set is used. When the program is invoked, this operation is performed on the name of the program, then the -p optional arguement, if any, and then any CPU statements. The last one selects the which subset of the registers the assembler will accept.

Register Architecture	Substrings
z8	86 z8 Z8
Universal Peripheral Controller	UPC upc 9

A.2 **Instructions**

A.2.1 ***Instruction_List***

Opcode	Syntax	Selection Criteria
ADC	'@' REGISTER ',' '#' expr	
ADC	REGISTER ',' '#' expr	
ADC	REGISTER ',' '@' REGISTER	
ADC	REGISTER ',' '@' REGISTER	SRCWORK+DSTWORK
ADC	REGISTER ',' REGISTER	
ADC	REGISTER ',' REGISTER	SRCWORK+DSTWORK
ADD	'@' REGISTER ',' '#' expr	
ADD	REGISTER ',' '#' expr	
ADD	REGISTER ',' '@' REGISTER	
ADD	REGISTER ',' '@' REGISTER	SRCWORK+DSTWORK
ADD	REGISTER ',' REGISTER	

Opcode	Syntax	Selection Criteria
ADD	REGISTER ',' REGISTER	SRCWORK+DSTWORK
AND	'@' REGISTER ',' '#' expr	
AND	REGISTER ',' '#' expr	
AND	REGISTER ',' '@' REGISTER	
AND	REGISTER ',' '@' REGISTER	SRCWORK+DSTWORK
AND	REGISTER ',' REGISTER	
AND	REGISTER ',' REGISTER	SRCWORK+DSTWORK
CALL	'@' REGISTER	DSTDBL
CALL	expr	
CCF		
CLR	'@' REGISTER	
CLR	REGISTER	
COM	'@' REGISTER	
COM	REGISTER	
CP	'@' REGISTER ',' '#' expr	
CP	REGISTER ',' '#' expr	
CP	REGISTER ',' '@' REGISTER	
CP	REGISTER ',' '@' REGISTER	SRCWORK+DSTWORK
CP	REGISTER ',' REGISTER	
CP	REGISTER ',' REGISTER	SRCWORK+DSTWORK
DA	'@' REGISTER	
DA	REGISTER	
DEC	'@' REGISTER	
DEC	REGISTER	
DECW	'@' REGISTER	
DECW	REGISTER	DSTDBL
DI		
DJNZ	REGISTER ',' expr	DSTWORK
EI		
INC	'@' REGISTER	
INC	REGISTER	
INC	REGISTER	DSTWORK
INCW	'@' REGISTER	
INCW	REGISTER	DSTDBL

Opcode	Syntax	Selection Criteria
IRET		
JP	'@' REGISTER	DSTDBL
JP	CONDITION ',' expr	
JP	expr	
JR	CONDITION ',' expr	
JR	expr	
LD	'@' REGISTER ',' '#' expr	
LD	'@' REGISTER ',' REGISTER	
LD	'@' REGISTER ',' REGISTER	DSTWORK+SRCWORK
LD	REGISTER ',' '#' expr	
LD	REGISTER ',' '#' expr	DSTWORK
LD	REGISTER ',' '@' REGISTER	
LD	REGISTER ',' '@' REGISTER	DSTWORK+SRCWORK
LD	REGISTER ',' REGISTER	
LD	REGISTER ',' REGISTER	DSTWORK
LD	REGISTER ',' REGISTER	SRCWORK
LD	REGISTER ',' expr '(' REGISTER ')'	DSTWORK+SRCWORK
LD	expr '(' REGISTER ')' ',' REGISTER	DSTWORK+SRCWORK
LDC	'@' REGISTER ',' REGISTER	DSTWORK+SRCWORK
LDC	REGISTER ',' '@' REGISTER	DSTWORK+SRCWORK
LDCI	'@' REGISTER ',' '@' REGISTER	DSTDBL+DSTWORK+SRCWORK
LDCI	'@' REGISTER ',' '@' REGISTER	SRCDBL+DSTWORK+SRCWORK
LDE	'@' REGISTER ',' REGISTER	DSTWORK+SRCWORK
LDE	REGISTER ',' '@' REGISTER	DSTWORK+SRCWORK
LDEI	'@' REGISTER ',' '@' REGISTER	DSTDBL+DSTWORK+SRCWORK
LDEI	'@' REGISTER ',' '@' REGISTER	SRCDBL+DSTWORK+SRCWORK
NOP		
OR	'@' REGISTER ',' '#' expr	
OR	REGISTER ',' '#' expr	
OR	REGISTER ',' '@' REGISTER	
OR	REGISTER ',' '@' REGISTER	SRCWORK+DSTWORK
OR	REGISTER ',' REGISTER	
OR	REGISTER ',' REGISTER	SRCWORK+DSTWORK
POP	'@' REGISTER	
POP	REGISTER	
PUSH	'@' REGISTER	
PUSH	REGISTER	

Opcode	Syntax	Selection Criteria
RCF		
RET		
RL	'@' REGISTER	
RL	REGISTER	
RLC	'@' REGISTER	
RLC	REGISTER	
RR	'@' REGISTER	
RR	REGISTER	
RRC	'@' REGISTER	
RRC	REGISTER	
SBC	'@' REGISTER ',' '#' expr	
SBC	REGISTER ',' '#' expr	
SBC	REGISTER ',' '@' REGISTER	
SBC	REGISTER ',' '@' REGISTER	SRCWORK+DSTWORK
SBC	REGISTER ',' REGISTER	
SBC	REGISTER ',' REGISTER	SRCWORK+DSTWORK
SCF		
SRA	'@' REGISTER	
SRA	REGISTER	
SRP	'#' expr	
SUB	'@' REGISTER ',' '#' expr	
SUB	REGISTER ',' '#' expr	
SUB	REGISTER ',' '@' REGISTER	
SUB	REGISTER ',' '@' REGISTER	SRCWORK+DSTWORK
SUB	REGISTER ',' REGISTER	
SUB	REGISTER ',' REGISTER	SRCWORK+DSTWORK
SWAP	'@' REGISTER	
SWAP	REGISTER	
TCM	'@' REGISTER ',' '#' expr	
TCM	REGISTER ',' '#' expr	
TCM	REGISTER ',' '@' REGISTER	
TCM	REGISTER ',' '@' REGISTER	SRCWORK+DSTWORK
TCM	REGISTER ',' REGISTER	
TCM	REGISTER ',' REGISTER	SRCWORK+DSTWORK
TM	'@' REGISTER ',' '#' expr	
TM	REGISTER ',' '#' expr	

Opcode	Syntax	Selection Criteria
TM	REGISTER ',' '@' REGISTER	
TM	REGISTER ',' '@' REGISTER	SRCWORK+DSTWORK
TM	REGISTER ',' REGISTER	
TM	REGISTER ',' REGISTER	SRCWORK+DSTWORK
XOR	'@' REGISTER ',' '#' expr	
XOR	REGISTER ',' '#' expr	
XOR	REGISTER ',' '@' REGISTER	
XOR	REGISTER ',' '@' REGISTER	SRCWORK+DSTWORK
XOR	REGISTER ',' REGISTER	
XOR	REGISTER ',' REGISTER	SRCWORK+DSTWORK

A.2.2 ***Selection_Criteria_Keywords***

DSTWORK	The instruction will use the short form if the register is in the working set.
DSTDBL	The instruction requires that the destination register be a double register.
SRCWORK	The instruction will use the short form if the register is in the working set.
SRCDBL	The instruction requires that the source register be a double register.

A.2.3 ***Apostrophes*** The apostrophes in the syntax field are a notation used for the parser generator and are not put in the assembler source statement.

A.3 **Notes**

A.3.1 ***Registers*** Registers are treated as symbols with a different syntactic type than memory addresses. This seperates symbols which would have the same numeric value into the different address spaces, register and memory. Using a symbol that is already defined as a Register as a statement label for an instruction or a SET/EQU will result in a "Syntax Error".

Registers are represented by the reserved symbols R0 thru R15 and RR0 thru RR14 for the working set registers. Other registers are defined with the REG and RREG pseudooperations.

Register Symbols must be surrounded by parenthesis when used in expressions. Converted register values are meaningless as ROM or Data memory addresses. Working set registers have a value between 0e0h and 0efh.

Example

```
ld r3, #(buffstart)
```

A.3.2 ***Data_Memory*** ROM and Data memory addresses are not differentiated.

A.3.3 ***Conditions*** Conditions are represented by the reserved symbols F, C, NC, Z, NZ, PL, MI, OV, NOV, EQ, NE, GE, LT, GT, LE, UGE, ULT, UGT, ULE

Note that the EQ, NE, GT, LE, GE, LT operators are not available in this assembler, the =, <>, >, <=, >=, < special character representations must be used.

A.3.4 ***Reserved_Symbols***

A.3.4.1 ***Machine_Dependent_Reserved_Symbols*** AND C DEFINED EQ F GE GT HIGH LE LOW LT MI MOD NC NE NOT NOV NZ OR OV PL R0 R1 R10 R11 R12 R13 R14 R15 R2 R3 R4 R5 R6 R7 R8 R9 RR0 RR10 RR12 RR14 RR2 RR4 RR6 RR8 SHL SHR UGE UGT ULE ULT XOR Z and defined high low mod not or shl shr xor

CONTENTS