FORTRAN Syntactic Analysis Specification

W. M. Waite

July 7, 1998

Abstract

This document defines the grammars used by X3.9-1978 (FORTRAN 77) and ISO/IEC 1539 (FORTRAN 90). It is a part of an Eli specification from which scanner/parsers for FORTRAN 77 and FORTRAN 90 can be generated. Generation of the parser is controlled by a FunnelWeb macro. The following selects the code for FORTRAN 77 analysis (`0` selects the code for FORTRAN 90 analysis):

Fortran77[1] ≡
`1`This macro is invoked in definitions 18 and 20.

Some symbols of the FORTRAN 77 grammar have been changed to their FORTRAN 90 counterparts, and changes necessary to remove ambiguity have been made. Cross references to both the FORTRAN 77 and FORTRAN 90 standards are included.

Additional rules specific to FORTRAN 90 can be found in a separate document.

This specification was created originally in January of 1987 by W. M. Waite and J. Hoffmann. Further development in the fall of 1989 by R. Jakob made it a basis for an analyzer for the "Force" extension of FORTRAN. Both of these versions only handled "nice looking" FORTRAN programs, because of weakness in the lexical analyzer. These restrictions were removed in the summer of 1993 by W. M. Waite, while he was a visiting researcher at the GMD in Berlin.

Contents

1 Context-Free Grammar

The grammar is structured according to Appendix F of X3.9-1978, and contains references to the rule numbers in ISO/IEC 1539. Cases in which strict adherence to Appendix F of X3.9-1978 would have either resulted in LALR(1) conflicts or compromised later semantic analysis are explained in subsequent sections. The rules for those cases are replaced here by macro calls giving the name of the section in which the change is explained.

1.1 Productions Common to FORTRAN 77 and FORTRAN 90

Productions Common to FORTRAN 77 and FORTRAN 90[2] ≡

```
% 1
% R201
xExecutableProgram:
  xProgramUnit / xExecutableProgram xProgramUnit .

% R202
xProgramUnit:
  xMainProgram /
  xFunctionSubprogram /
  xSubroutineSubprogram /
  xBlockDataSubprogram .

% 2
% R1101
xMainProgram:
  xMainRange /
  xProgramStmt xMainRange .
xMainRange:
  xBody xEndProgramStmt /
  xEndProgramStmt .
xEndProgramStmt: xLblDef 'end' xEOS .

xBody:
  xBodyConstruct / xBody xBodyConstruct .
xBodyConstruct:
  xSpecificationPartConstruct / xExecutableConstruct .

% R204
xSpecificationPartConstruct:
  xImplicitStmt / xParameterStmt / xFormatStmt / xEntryStmt /
  xDeclarationConstruct .
```

```
% R207
xDeclarationConstruct:
  xTypeDeclarationStmt / xSpecificationStmt .

% 3
% R1215
xFunctionSubprogram:
  xLblDef xFunctionPrefix xFunctionName xFunctionRange .
xFunctionRange:
  xFunctionParList xEOS xBody xEndFunctionStmt /
  xFunctionParList xEOS xEndFunctionStmt .
xEndFunctionStmt: xLblDef 'end' xEOS .

% 4
% R1219
xSubroutineSubprogram:
  xLblDef 'subroutine' xSubroutineName xSubroutineRange .
xSubroutineRange:
  xSubroutineParList xEOS xBody xEndSubroutineStmt /
  xSubroutineParList xEOS xEndSubroutineStmt .
xEndSubroutineStmt: xLblDef 'end' xEOS .

% 5
% R1110
xBlockDataSubprogram:
  xBlockDataStmt xBlockDataBody xEndBlockDataStmt /
  xBlockDataStmt xEndBlockDataStmt .
xEndBlockDataStmt: xLblDef 'end' xEOS .

xBlockDataBody:
  xBlockDataBodyConstruct / xBlockDataBody xBlockDataBodyConstruct .
xBlockDataBodyConstruct:
  xSpecificationPartConstruct .

% 6
xSpecificationStmt:
  xCommonStmt /
  xDataStmt /
  xDimensionStmt /
  xEquivalenceStmt /
  xExternalStmt /
  xIntrinsicStmt /
  xSaveStmt .
```

```
% 7
xExecutionPartConstruct:
  xExecutableConstruct / xFormatStmt / xDataStmt / xEntryStmt .

xExecutableConstruct:
  xActionStmt /
  xDoConstruct /
  xIfConstruct .

xActionStmt:
  xArithmeticIfStmt /
  xAssignmentStmt / xAssignStmt /
  xBackspaceStmt /
  xCallStmt /
  xCloseStmt /
  xContinueStmt /
  xEndfileStmt /
  xGotoStmt / xComputedGotoStmt / xAssignedGotoStmt /
  xIfStmt /
  xInquireStmt /
  xOpenStmt /
  xPauseStmt /
  xPrintStmt /
  xReadStmt /
  xReturnStmt /
  xRewindStmt /
  xStmtFunctionStmt /
  xStopStmt /
  xWriteStmt .

% 8
xProgramStmt:
  xLblDef 'program' xProgramName xEOS .

% 9 see Entry Statement

% 10
/* Must be split on semantic grounds, due to the different scopes for the
 * function name and the dummy parameters (if any).  See 3
 *
 * xFunctionStmt:
 *   xLblDef xFunctionPrefix xName xFunctionParList xEOS .
 */
xFunctionPrefix: 'function' / xTypeSpec 'function' .
```

```
xFunctionParList: / '(' xFunctionPars ')' .
xFunctionPars: / xFunctionPar / xFunctionPars ',' xFunctionPar .
xFunctionPar: xDummyArgName .

% 11 parsed as 13, distinguished semantically

% 12
/* Must be split on semantic grounds, due to the different scopes for the
 * subroutine name and the dummy parameters (if any).  See 4
 *
 * xSubroutineStmt:
 *   xLblDef 'subroutine' xName xSubroutineParList xEOS.
 */
xSubroutineParList: / '(' xSubroutinePars ')' .
xSubroutinePars: / xSubroutinePar / xSubroutinePars ',' xSubroutinePar .
xSubroutinePar: xDummyArgName / '*' .

% 13
```
Entry Statement[4]

```
% 14
xBlockDataStmt:
  xLblDef 'blockdata' xBlockDataName xEOS /
  xLblDef 'blockdata' xEOS .

% 15
% R525
xDimensionStmt:
  xLblDef 'dimension' xArrayDeclaratorList xEOS .

% 16
xArrayDeclaratorList:
  xArrayDeclarator / xArrayDeclaratorList ',' xArrayDeclarator .
xArrayDeclarator: xVariableName '(' xArraySpec ')' .

% R512
xArraySpec: xExplicitShapeSpecList / xAssumedSizeSpec .

% R513
xExplicitShapeSpecList:
  xExplicitShapeSpec / xExplicitShapeSpecList ',' xExplicitShapeSpec .
xExplicitShapeSpec: xLowerBound ':' xUpperBound / xUpperBound .
```

```
% R514
xLowerBound: Specification Expression[5] .

% R515
xUpperBound: Specification Expression[5] .

% R518
xAssumedSizeSpec:
  '*' /
  xLowerBound ':' '*' /
  xExplicitShapeSpecList ',' '*' /
  xExplicitShapeSpecList ',' xLowerBound ':' '*' .

% 17
% R545
xEquivalenceStmt:
  xLblDef 'equivalence' xEquivalenceSetList xEOS .

% R546
xEquivalenceSetList:
  xEquivalenceSet / xEquivalenceSetList ',' xEquivalenceSet .
xEquivalenceSet: '(' xEquivalenceObject ',' xEquivalenceObjectList ')' .

% 18
% R547
xEquivalenceObjectList:
  xEquivalenceObject / xEquivalenceObjectList ',' xEquivalenceObject .
xEquivalenceObject: Equivalence Entity[6] .

% 19
% R548
xCommonStmt:
  xLblDef 'common' xComlist xEOS .
xComlist:
  xCommonBlockObject /
  xComblock xCommonBlockObject /
  xComlist ',' xCommonBlockObject /
  xComlist xComblock xCommonBlockObject /
  xComlist ',' xComblock xCommonBlockObject .
xComblock: '/' '/' / '/' xCommonBlockName '/' .
xCommonBlockObject: xVariableName / xArrayDeclarator .

% 20
% R501
```

```
xTypeDeclarationStmt:
  xLblDef xTypeSpec xEntityDeclList xEOS .

% R502
xTypeSpec:
  'integer' /
  'real' /
  'doubleprecision' /
  'complex' /
  'logical' /
  'character' /
  'character' xLengthSelector .

% R504
xEntityDeclList: xEntityDecl / xEntityDeclList ',' xEntityDecl .
xEntityDecl:
  xObjectName /
  xObjectName '(' xArraySpec ')' /
  xObjectName '*' xCharLength /
  xObjectName '(' xArraySpec ')' '*' xCharLength .

% R507
xLengthSelector: '*' xCharLength .

% 21
% R540
xImplicitStmt:
  xLblDef 'implicit' xImplicitSpecList xEOS .

% R541
xImplicitSpecList: xImplicitSpec / xImplicitSpecList ',' xImplicitSpec .
```
xImplicitSpec: xTypeSpec *Letter Specifications*[7] .

```
% 22
% R508
```
xCharLength: '(' xTypeParamValue ')' / *Scalar Integer Literal Constant*[8] .

```
% R509
```
xTypeParamValue: *Specification Expression*[5] / '*' .

```
% 23
% R538
xParameterStmt:
  xLblDef 'parameter' '(' xNamedConstantDefList ')' xEOS .
```

```
% R539
xNamedConstantDefList:
  xNamedConstantDef / xNamedConstantDefList ',' xNamedConstantDef .
xNamedConstantDef: xNamedConstant '=' Constant Expression[9] .

% R307
xNamedConstant: xIdent .
xNamedConstantUse: xIdent .

% 24
% R1207
xExternalStmt:
  xLblDef 'external' xExternalNameList xEOS .
xExternalNameList: xExternalName / xExternalNameList ',' xExternalName .

% 25
% R1208
xIntrinsicStmt:
  xLblDef 'intrinsic' xIntrinsicList xEOS .
xIntrinsicList:
  xIntrinsicProcedureName / xIntrinsicList ',' xIntrinsicProcedureName .

% 26
% R523
xSaveStmt:
  xLblDef 'save' xEOS /
  xLblDef 'save' xSavedEntityList xEOS .

% R524
xSavedEntityList: xSavedEntity / xSavedEntityList ',' xSavedEntity .
xSavedEntity: xVariableName / xSavedCommonBlock .
xSavedCommonBlock: '/' xCommonBlockName '/' .

% 27
% R529
xDataStmt:
  xLblDef 'data' xDatalist xEOS .
xDatalist: xDataStmtSet / xDatalist xDataStmtSet / xDatalist ',' xDataStmtSet .

% R530
xDataStmtSet: xDataStmtObjectList '/' xDataStmtValueList '/' .

% R531
```

```
xDataStmtObjectList: xDataStmtObject / xDataStmtObjectList ',' xDataStmtObject .
xDataStmtObject: xVariable / xDataImpliedDo .

% R532
xDataStmtValueList: xDataStmtValue / xDataStmtValueList ',' xDataStmtValue .
xDataStmtValue:
  xConstant /
  Scalar Integer Literal Constant[8] '*' xConstant /
  xNamedConstantUse '*' xConstant .

% 28
% R535
xDataImpliedDo:
  '(' xDataIDoObjectList ',' xImpliedDoVariable '=' xExpr ',' xExpr ')' /
  '(' xDataIDoObjectList ',' xImpliedDoVariable '=' xExpr ',' xExpr
    ',' xExpr ')' .

% R536
xDataIDoObjectList: xDataIDoObject / xDataIDoObjectList ',' xDataIDoObject .
xDataIDoObject: xArrayElement / xDataImpliedDo .

% 29
Assignment Statement[13]

% 30 see 31-33

% 31
xGotoStmt:
  xLblDef GoToKw xLblRef xEOS .

GoToKw: 'goto' .

% 32
xComputedGotoStmt:
  xLblDef GoToKw '(' xLblRefList ')' Integer Expression[10] xEOS /
  xLblDef GoToKw '(' xLblRefList ')' xCommaExp xEOS .
xCommaExp: ',' Integer Expression[10] .
xLblRefList: xLblRef / xLblRefList ',' xLblRef .
xLblRef: xLabel .

% 33
% R839
xAssignedGotoStmt:
  xLblDef GoToKw xVariableName xEOS /
```

```
  xLblDef GoToKw xVariableName '(' xLblRefList ')' xEOS /
  xLblDef GoToKw xVariableComma '(' xLblRefList ')' xEOS .
xVariableComma: xVariableName ',' .

% 34
% R840
xArithmeticIfStmt:
  xLblDef 'if' '(' Int-Real-Dp Expression[11] ')'
         xLblRef ',' xLblRef ',' xLblRef xEOS .

% 35
% R807
xIfStmt:
  xLblDef 'if' '(' Scalar Logical Expression[12] ')' xActionStmt .

% R802
xIfConstruct:
  xIfThenStmt xThenPart .
xThenPart:
  xEndIfStmt / xConditionalBody xEndIfStmt /
  xElseIfConstruct / xConditionalBody xElseIfConstruct /
  xElseConstruct / xConditionalBody xElseConstruct .

xElseIfConstruct:
  xElseIfStmt xThenPart .

xElseConstruct:
  xElseStmt xElsePart .
xElsePart:
  xEndIfStmt /
  xConditionalBody xEndIfStmt .

xConditionalBody:
  xExecutionPartConstruct / xConditionalBody xExecutionPartConstruct .

% 36
% R803
xIfThenStmt:
  xLblDef 'if' '(' Scalar Logical Expression[12] ')' 'then' xEOS .

% 37
% R804
xElseIfStmt:
  xLblDef 'elseif' '(' Scalar Logical Expression[12] ')' 'then' xEOS .
```

```
% 38
% R805
xElseStmt:
  xLblDef 'else' xEOS .

% 39
% R806
xEndIfStmt:
  xLblDef 'endif' xEOS.

% 40
% R818
xDoConstruct: xLabelDoStmt .

% R819
xLabelDoStmt:
  xLblDef 'do' xLblRef xCommaLoopControl xEOS .
xCommaLoopControl: ',' xLoopControl / xLoopControl .

% R821
xLoopControl:
  xVariableName '='
    Int-Real-Dp Expression[11] ',' Int-Real-Dp Expression[11] /
  xVariableName '='
    Int-Real-Dp Expression[11] ',' Int-Real-Dp Expression[11] ','
    Int-Real-Dp Expression[11] .

% 41
% R841
xContinueStmt:
  xLblDef 'continue' xEOS .

% 42
% R842
xStopStmt:
  xLblDef 'stop' xEOS /
  xLblDef 'stop' xIcon xEOS /
  xLblDef 'stop' xScon xEOS .

% 43
% R844
xPauseStmt:
  xLblDef 'pause' xEOS /
```

```
  xLblDef 'pause' xIcon xEOS /
  xLblDef 'pause' xScon xEOS .

% 44
xWriteStmt:
  xLblDef 'write' '(' xIoControlSpecList ')' xOutputItemList xEOS /
  xLblDef 'write' '(' xIoControlSpecList ')' xEOS .

% 45
xReadStmt:
  xLblDef 'read' xRdCtlSpec xInputItemList xEOS /
  xLblDef 'read' xRdCtlSpec xEOS /
  xLblDef 'read' xRdFmtId ',' xInputItemList xEOS /
  xLblDef 'read' xRdFmtId xEOS .
xRdCtlSpec: xRdUnitId / '(' xRdIoCtlSpecList ')' .
xRdUnitId: '(' xUFExpr ')' / '(' '*' ')' .
xRdIoCtlSpecList:
  xUnitIdentifier ',' xIoControlSpec /
  xUnitIdentifier ',' xFormatIdentifier /
  xIoControlSpec /
  xRdIoCtlSpecList ',' xIoControlSpec .
xRdFmtId:
  xLblRef / '*' / xCOperand /
  xCOperand xConcatOp xCPrimary /
  xRdFmtIdExpr xConcatOp xCPrimary .
xRdFmtIdExpr: '(' xUFExpr ')' .

% 46
xPrintStmt:
  xLblDef 'print' xFormatIdentifier ',' xOutputItemList xEOS /
  xLblDef 'print' xFormatIdentifier xEOS .

% 47
xIoControlSpecList:
  xUnitIdentifier $',' /
  xUnitIdentifier ',' xFormatIdentifier /
  xUnitIdentifier ',' xIoControlSpec /
  xIoControlSpec /
  xIoControlSpecList ',' xIoControlSpec .

% R912
xIoControlSpec:
  'fmt=' xFormatIdentifier /
  'unit=' xUnitIdentifier /
```

```
  'rec=' xExpr /
  'end=' xLblRef /
  'err=' xLblRef /
  'iostat=' xScalarVariable .

% 48
% R914
xInputItemList: xInputItem / xInputItemList ',' xInputItem .
xInputItem: xVariable / xInputImpliedDo .

% R915
xOutputItemList: xExpr / xOutputItemList1 .
xOutputItemList1:
  xExpr ',' xExpr /
  xExpr ',' xOutputImpliedDo /
  xOutputImpliedDo /
  xOutputItemList1 ',' xExpr /
  xOutputItemList1 ',' xOutputImpliedDo .

% 49
% R916
xInputImpliedDo:
  '(' xInputItemList ',' xImpliedDoVariable '=' xExpr ',' xExpr ')' /
  '(' xInputItemList ',' xImpliedDoVariable '=' xExpr ',' xExpr ',' xExpr ')' .

xOutputImpliedDo:
  '(' xExpr ',' xImpliedDoVariable '=' xExpr ',' xExpr ')' /
  '(' xExpr ',' xImpliedDoVariable '=' xExpr ',' xExpr ',' xExpr ')' /
  '(' xOutputItemList1 ',' xImpliedDoVariable '=' xExpr ',' xExpr ')' /
  '(' xOutputItemList1 ',' xImpliedDoVariable '=' xExpr ',' xExpr
    ',' xExpr ')' .


% 50
xOpenStmt:
  xLblDef 'open' '(' xConnectSpecList ')' xEOS .

% R905
xConnectSpecList:
  xUnitIdentifier /
  xConnectSpec / xConnectSpecList ',' xConnectSpec .
xConnectSpec:
  'unit=' xUnitIdentifier /
  'err=' xLblRef /
```

```
  'file=' xCExpr /
  'status=' xCExpr /
  'access=' xCExpr /
  'form=' xCExpr /
  'recl=' xExpr /
  'blank=' xCExpr /
  'iostat=' xScalarVariable .

% 51
xCloseStmt:
  xLblDef 'close' '(' xCloseSpecList ')' xEOS .

% R908
xCloseSpecList:
  xUnitIdentifier /
  xCloseSpec / xCloseSpecList ',' xCloseSpec .
xCloseSpec:
  'unit=' xUnitIdentifier /
  'err=' xLblRef /
  'status=' xCExpr /
  'iostat=' xScalarVariable .

% 52
% R923
xInquireStmt:
  xLblDef 'inquire' '(' xInquireSpecList ')' xEOS .

% R924
xInquireSpecList:
  xUnitIdentifier /
  xInquireSpec / xInquireSpecList ',' xInquireSpec .
xInquireSpec:
  'unit=' xUnitIdentifier /
  'file=' xCExpr /
  'err=' xLblRef /
  'iostat=' xScalarVariable /
  'exist=' xScalarVariable /
  'opened=' xScalarVariable /
  'number=' xScalarVariable /
  'named=' xScalarVariable /
  'name=' xScalarVariable /
  'access=' xScalarVariable /
  'sequential=' xScalarVariable /
  'direct=' xScalarVariable /
```

```
  'form=' xScalarVariable /
  'formatted=' xScalarVariable /
  'unformatted=' xScalarVariable /
  'recl=' xExpr /
  'nextrec=' xScalarVariable /
  'blank=' xScalarVariable .

% 53
% R919
xBackspaceStmt:
  xLblDef 'backspace' xUnitIdentifier xEOS /
  xLblDef 'backspace' '(' xPositionSpecList ')' xEOS .

% 54
% R920
xEndfileStmt:
  xLblDef 'endfile' xUnitIdentifier xEOS /
  xLblDef 'endfile' '(' xPositionSpecList ')' xEOS .

% 55
% R921
xRewindStmt:
  xLblDef 'rewind' xUnitIdentifier xEOS /
  xLblDef 'rewind' '(' xPositionSpecList ')' xEOS .

% R922
xPositionSpecList:
  xUnitIdentifier ',' xPositionSpec / xPositionSpec /
  xPositionSpecList ',' xPositionSpec .
xPositionSpec:
  'unit=' xUnitIdentifier /
  'err=' xLblRef /
  'iostat=' xScalarVariable .

% 56
xUnitIdentifier: xUFExpr / '*' .

% 57
xFormatIdentifier: xLblRef / xCExpr / '*' .

% 58-59
xFormatStmt:
  xLblDef 'format' '(' xFmtSpec ')' xEOS .
```

```
% 60
xFmtSpec:
  xFormatedit / xFormatsep / xFormatsep xFormatedit /
  xFmtSpec xFormatsep /
  xFmtSpec xFormatsep xFormatedit /
  xFmtSpec ',' xFormatedit /
  xFmtSpec ',' xFormatsep /
  xFmtSpec ',' xFormatsep xFormatedit .
xFormatedit:
  xEditElement / xIcon xEditElement / xXcon /
  xPcon / xPcon xEditElement / xPcon xIcon xEditElement .
xEditElement:
  xFcon / xScon / xHcon / xIdent / '(' xFmtSpec ')' .
xFormatsep: '/' / ':' .

% 61 recognized by the lexical analyzer as xIcon

% 62 recognized by the lexical analyzer in the context of xIdent or xFcon

% 63 recognized by the lexical analyzer in the context of xFcon

% 64 recognized by the lexical analyzer in the context of xHcon or xXcon

% 65 recognized by the lexical analyzer in the context of xIdent

% 66-67 recognized by the lexical analyzer in the context of xFcon

% 68 recognized by the lexical analyzer in the context of xPcon

% 69 recognized by the lexical analyzer in the context of xScon or xHcon

% 70
/* This may turn out to be an assignment statement, but the form given here
 * allows for name analysis in the case that it actually IS a statement
 * function definition.
 */
xStmtFunctionStmt: xLblDef xName xStmtFunctionRange .
xStmtFunctionRange: '(' ')' '=' xExpr xEOS .
xStmtFunctionRange: '(' xSFDummyArgNameList ')' '=' xExpr xEOS .
xSFDummyArgNameList:
  xSFDummyArgName / xSFDummyArgNameList ',' xSFDummyArgName .

% 71
xCallStmt:
```

```
  xLblDef 'call' xSubroutineNameUse xEOS /
  xLblDef 'call' xSubroutineNameUse '(' xSubroutineArgList ')' xEOS .
xSubroutineArgList: / xSubroutineArg / xSubroutineArgList ',' xSubroutineArg .
xSubroutineArg: xExpr / xHcon / '*' xLblRef .

% 72
xReturnStmt:
  xLblDef 'return' xEOS /
  xLblDef 'return' xExpr xEOS .

% 73
xFunctionReference: xName '(' ')' .

xComplexDataRef:
  xName  '(' xSectionSubscriptList ')' /
  xComplexDataRef '(' xSectionSubscriptList ')' .
xSectionSubscriptList:
  xSectionSubscript / xSectionSubscriptList ',' xSectionSubscript .
xSectionSubscript: xExpr / xSubscriptTriplet .

% 74-75
% R723
xExpr: xLevel5Expr .

% 76-81
% R701
xPrimary:
  xUnsignedArithmeticConstant /
  xName /
  xComplexDataRef /
  xFunctionReference /
  '(' xExpr ')' .

% R703
xLevel1Expr: xPrimary .

% R705
xMultOperand: xLevel1Expr [xPowerOp xMultOperand] .

% R706
xAddOperand: [xAddOperand xMultOp] xMultOperand .

% R707
xLevel2Expr:
```

```
  [xLevel2Expr xAddOp] xAddOperand /
  xSign xAddOperand .   % We need to distinguish unary operators

% R708
xPowerOp: '**' .

% R709
xMultOp: '*' / '/' .

% R710
xAddOp: '+' / '-' .
xSign: '+' / '-' .

xUFExpr: xUFTerm / xSign xUFTerm / xUFExpr xAddOp xUFTerm .
xUFTerm: xUFFactor / xUFTerm xMultOp xUFFactor / xUFTerm xConcatOp xUFPrimary .
xUFFactor: xUFPrimary / xUFPrimary xPowerOp xUFFactor .
xUFPrimary:
  xIcon /
  xScon /
  xName /
  xFunctionReference /
  xComplexDataRef /
  '(' xUFExpr ')' .

% 82,83
xCExpr: [xCExpr xConcatOp] xCPrimary .
xCPrimary: xCOperand / '(' xCExpr ')' .
xCOperand:
  xScon /
  xName /
  xComplexDataRef /
  xFunctionReference .

xPrimary:
  xScon .

% R711
xLevel3Expr: [xLevel3Expr xConcatOp] xLevel2Expr .

% R712
xConcatOp: '//' .

% 84,85
% R715
```

```
xAndOperand: [xNotOp] xLevel4Expr .

% R716
xOrOperand: [xOrOperand xAndOp] xAndOperand .

% R717
xEquivOperand: [xEquivOperand xOrOp] xOrOperand .

% R718
xLevel5Expr: [xLevel5Expr xEquivOp] xEquivOperand .

% R719
xNotOp: '.not.' .

% R720
xAndOp: '.and.' .

% R721
xOrOp: '.or.' .

% R722
xEquivOp: '.eqv.' / '.neqv.' .

xPrimary: xLogicalConstant .

% 86
xLevel4Expr: [xLevel3Expr xRelOp] xLevel3Expr .

% 87
xRelOp: '.eq.' / '.ne.' / '.lt.' / '.le.' / '.gt.' / '.ge.' .

% 88
xArrayElement: xVariableName '(' xSectionSubscriptList ')' .

% 89
xSubstringRange: '(' xSubscriptTriplet ')' .
xSubscriptTriplet:
  ':' /
  ':' xExpr /
  xExpr ':' /
  xExpr ':' xExpr .

% 90-99
xName: xIdent .
```

```
% 100
xConstant:
  xNamedConstantUse /
  xUnsignedArithmeticConstant /
  '+' xUnsignedArithmeticConstant /
  '-' xUnsignedArithmeticConstant /
  xScon / xHcon /
  xLogicalConstant .

% 101
xUnsignedArithmeticConstant:
  xIcon / xRcon / xDcon / xComplexConst .

% 102 recognized by the lexical analyzer as xIcon

% 103 parsed as 102, distinguished semantically

% 104 recognized by the lexical analyzer in the context of 68, 105 or 106

% 105 recognized by the lexical analyzer as xRcon

% 106 recognized by the lexical analyzer as xDcon

% 107
xComplexConst: '(' xExpr ',' xExpr ')' .

% 108
xLogicalConstant: '.true.' / '.false.' .

% 109 recognized by the lexical analyzer as xScon

% 110
xLabel: xIcon .

% 111-116 are components of symbols recognized by the lexical analyzer

/* Nonterminal symbols that are not defined in the standard
 */

xBlockDataName: xIdent .
xCommonBlockName: xIdent .
xDummyArgName: xIdent .
xEntryName: xIdent .
```

```
xExternalName: xIdent .
xFunctionName: xIdent .
xImpliedDoVariable: xIdent .
xIntrinsicProcedureName: xIdent .
xObjectName: xIdent .
xProgramName: xIdent .
xSFDummyArgName: xIdent .
xSFVarName: xIdent $',' $')' .
xSubroutineName: xIdent .
xSubroutineNameUse: xIdent .
xVariableName: xIdent .

xScalarVariable: xVariableName / xArrayElement .
xVariable:
  xVariableName /
  xVariableName '(' xSubscriptList ')' /
  xVariableName xSubstringRange /
  xVariableName '(' xSubscriptList ')' xSubstringRange .
xSubscriptList:
  xSubscript / xSubscriptList ',' xSubscript .
xSubscript: xExpr .

xLblDef: / xLabel .
```

This macro is invoked in definition 18.

1.2 Productions Specific to FORTRAN 90

The grammar is structured according to ISO/IEC 1539. It only gives those rules that are specific to FORTRAN 90, and contains references to the rule numbers in Appendix F of X3.9-1978 for rules that are common to FORTRAN 77 and FORTRAN 90. Cases in which strict adherence to ISO/IEC 1539 would have either resulted in LALR(1) conflicts or compromised later semantic analysis are explained in subsequent sections. The rules for those cases are replaced here by macro calls giving the name of the section in which the change is explained.

Productions Specific to FORTRAN 90[3] ≡

```
% R201 see 1

% R202 see 1
xProgramUnit: xModule .

% R203 chain rule omitted

% R204 see 2
```

```
xSpecificationPartConstruct: xUseStmt .

% R205 see 2

% R206 see 2

% R207 see 2
xDeclarationConstruct: xDerivedTypeDef / xInterfaceBlock .

% R208 see 2

% R209 see 2

% R210
xBodyPlusInternals:
  xBody xContainsStmt xInternalSubprogram /
  xContainsStmt xInternalSubprogram /
  xBodyPlusInternals xInternalSubprogram .

% R211
xInternalSubprogram:
  xFunctionSubprogram / xSubroutineSubprogram .

% R212
xModuleSubprogramPartConstruct:
  xContainsStmt / xModuleSubprogram .

% R213
xModuleSubprogram:
  xFunctionSubprogram / xSubroutineSubprogram .

% R214 see 6
xSpecificationStmt:
  xAccessStmt /
  xAllocatableStmt /
  xIntentStmt /
  xNamelistStmt /
  xOptionalStmt /
  xPointerStmt /
  xTargetStmt .

% R215 see 7
xExecutableConstruct:
  xCaseConstruct /
```

```
  xWhereConstruct /
  xEndDoStmt .          /* see the note on R818 */

% R216 see 7
xActionStmt:
  xAllocateStmt /
  xCycleStmt /
  xDeallocateStmt /
  xExitStmt /
  xNullifyStmt /
  xPointerAssignmentStmt /
  xWhereStmt.

% R301-R304 are components of symbols recognized by the lexical analyzer

% R305 chain rules deleted

% R306 recognized by the lexical analyzer

% R307 see 23

% R308 chain rule deleted

% R309 chain rule deleted

% R310 recognized semantically

% R311
xDefinedOperator:
  xDop /
  xPowerOp / xMultOp / xAddOp / xRelOp / xNotOp / xAndOp / xOrOp / xEquivOp .

% R401 recognized by the lexical analyzer in the context of R413

% R402 recognized by the lexical analyzer in the context of R413, xIcon

% R404 see 101
xUnsignedArithmeticConstant: xIcon '_' xKindParam .

% R405
xKindParam: xIcon / xNamedConstantUse .

% R406 see 100
```

```
% R407
xBozLiteralConstant: xBcon / xOcon / xZcon .

% R408 recognized by the lexical analyzer as xBcon

% R409 recognized by the lexical analyzer as xOcon

% R410 recognized by the lexical analyzer as xZcon

% R411 component of a symbol recognized by the lexical analyzer

% R412 see 100

% R413 see 100
xUnsignedArithmeticConstant: xRcon '_' xKindParam / xDcon '_' xKindParam .

% R414-R416 components of symbols recognized by the lexical analyzer

% R417 see 107

% R418 chain rule deleted

% R419 chain rule deleted

% R420 see 100
xConstant:
  xIcon '_' xScon / xNamedConstantUse '_' xScon .

% R421 see 108
xLogicalConstant:
  '.true.' '_' xKindParam / '.false.' '_' xKindParam .

% R422
xDerivedTypeDef:
  xDerivedTypeStmt xDerivedTypeBody xEndTypeStmt .
xDerivedTypeBody:
  xDerivedTypeBodyConstruct / xDerivedTypeBody xDerivedTypeBodyConstruct .
xDerivedTypeBodyConstruct: xPrivateSequenceStmt / xComponentDefStmt .

% R423
xPrivateSequenceStmt:
  xLblDef 'private' xEOS /
  xLblDef 'sequence' xEOS .
```

```
% R424
xDerivedTypeStmt:
  xLblDef 'type' xTypeName xEOS /
  xLblDef 'type' ':' ':' xTypeName xEOS /
  xLblDef 'type' ',' xAccessSpec ':' ':' xTypeName xEOS .

% R425
xEndTypeStmt:
  xLblDef 'endtype' xTypeName xEOS /
  xLblDef 'endtype' xEOS /
  xLblDef 'end' 'type' xTypeName xEOS /
  xLblDef 'end' 'type' xEOS .

% R426
xComponentDefStmt:
  xLblDef xTypeSpec ',' xComponentAttrSpecList ':' ':' xComponentDeclList xEOS /
  xLblDef xTypeSpec ':' ':' xComponentDeclList xEOS /
  xLblDef xTypeSpec xComponentDeclList xEOS .

% R427
xComponentAttrSpecList:
  xComponentAttrSpec / xComponentAttrSpecList ',' xComponentAttrSpec .
xComponentAttrSpec:
  'pointer' / 'dimension' '(' xComponentArraySpec ')' .

% R428
xComponentArraySpec: xExplicitShapeSpecList / xDeferredShapeSpecList .

% R429
xComponentDeclList:
  xComponentDecl / xComponentDeclList ',' xComponentDecl .
xComponentDecl:
  xComponentName '(' xComponentArraySpec ')' '*' xCharLength /
  xComponentName '(' xComponentArraySpec ')' /
  xComponentName '*' xCharLength /
  xComponentName .

% R430
xStructureConstructor: xTypeName '(' xExprList ')' .
xExprList: xExpr / xExprList ',' xExpr .

% R431
xArrayConstructor: '(/' xAcValueList '/)' .
```

```
% R432
xAcValueList: xExpr / xAcValueList1 .
xAcValueList1:
  xExpr ',' xExpr /
  xExpr ',' xAcImpliedDo /
  xAcImpliedDo /
  xAcValueList1 ',' xExpr /
  xAcValueList1 ',' xAcImpliedDo .

% R433
xAcImpliedDo:
  '(' xExpr ',' xImpliedDoVariable '=' xExpr ',' xExpr ')' /
  '(' xExpr ',' xImpliedDoVariable '=' xExpr ',' xExpr ',' xExpr ')' /
  '(' xAcImpliedDo ',' xImpliedDoVariable '=' xExpr ',' xExpr ')' /
  '(' xAcImpliedDo ',' xImpliedDoVariable '=' xExpr ',' xExpr ',' xExpr ')' .

% R434 chain rule deleted

% R435 chain rule deleted

% R501 see 20
xTypeDeclarationStmt:
  xLblDef xTypeSpec xAttrSpecSeq ':' ':' xEntityDeclList xEOS /
  xLblDef xTypeSpec ':' ':' xEntityDeclList xEOS .
xAttrSpecSeq: ',' xAttrSpec / xAttrSpecSeq ',' xAttrSpec .

% R502 see 20
xTypeSpec:
  'integer' xKindSelector /
  'real' xKindSelector /
  'double' 'precision' /
  'complex' xKindSelector /
  'character' xCharSelector /
  'logical' xKindSelector /
  'type' '(' xTypeName ')' .

% R503
xAttrSpec:
  'parameter' /
  xAccessSpec /
  'allocatable' /
  'dimension' '(' xArraySpec ')' /
  'external' /
  'intent' '(' xIntentSpec ')' /
```

```
  'intrinsic' /
  'optional' /
  'pointer' /
  'save' /
  'target' .

% R504 see 20
xEntityDecl:
  xObjectName '=' xExpr /
  xObjectName '(' xArraySpec ')' '=' xExpr /
  xObjectName '*' xCharLength '=' xExpr /
  xObjectName '*' xCharLength '(' xArraySpec ')' '=' xExpr .

% R505
xKindSelector: '(' 'kind=' xExpr ')' / '(' xExpr ')' .

% R506 see 20
% R507 see 20
xCharSelector:
  '(' 'len=' xTypeParamValue ',' 'kind=' xExpr ')' /
  '(' 'len=' xTypeParamValue ',' xExpr ')' /
  '(' 'len=' xTypeParamValue ')' /
  '(' 'kind=' xExpr ')' /
  '(' xExpr ')' .

% R508 see 22

% R509 see 22

% R510
xAccessSpec:
  'public' / 'private' .

% R511
xIntentSpec:
  'in' / 'out' / 'inout' .

% R512 see 16
xArraySpec:
   xAssumedShapeSpecList / xDeferredShapeSpecList .
xAssumedShapeSpecList:
  xLowerBound ':' /
  xDeferredShapeSpecList ',' xLowerBound ':' /
  xAssumedShapeSpecList ',' xAssumedShapeSpec .
```

```
% R513 see 16

% R514 see 16

% R515 see 16

% R516
xAssumedShapeSpec:
  xLowerBound ':' / ':' .

% R517
xDeferredShapeSpecList:
  xDeferredShapeSpec / xDeferredShapeSpecList ',' xDeferredShapeSpec .
xDeferredShapeSpec: ':' .

% R518 see 16

% R519
xIntentStmt:
  xLblDef 'intent' '(' xIntentSpec ')' ':' ':' xIntentParList xEOS /
  xLblDef 'intent' '(' xIntentSpec ')' xIntentParList xEOS .
xIntentParList: xIntentPar / xIntentParList ',' xIntentPar .
xIntentPar: xDummyArgName .

% R520
xOptionalStmt:
  xLblDef 'optional' ':' ':' xOptionalParList xEOS /
  xLblDef 'optional' xOptionalParList xEOS .
xOptionalParList: xOptionalPar / xOptionalParList ',' xOptionalPar .
xOptionalPar: xDummyArgName .

% R521
xAccessStmt:
  xLblDef xAccessSpec ':' ':' xAccessIdList xEOS /
  xLblDef xAccessSpec xAccessIdList xEOS /
  xLblDef xAccessSpec xEOS .

% R522
xAccessIdList: xAccessId / xAccessIdList ',' xAccessId .
xAccessId: xGenericName / xGenericSpec .

% R523 see 26
xSaveStmt:
```

```
  xLblDef 'save' ':' ':' xSavedEntityList xEOS .

% R524 see 26

% R525 see 15
xDimensionStmt:
  xLblDef 'dimension' ':' ':' xArrayDeclaratorList xEOS .

% R526
xAllocatableStmt:
  xLblDef 'allocatable' ':' ':' xArrayAllocationList xEOS /
  xLblDef 'allocatable' xArrayAllocationList xEOS .
xArrayAllocationList:
  xArrayAllocation / xArrayAllocationList ',' xArrayAllocation .
xArrayAllocation:
  xArrayName / xArrayName '(' xDeferredShapeSpecList ')' .

% R527
xPointerStmt:
  xLblDef 'pointer' ':' ':' xPointerStmtObjectList xEOS /
  xLblDef 'pointer' xPointerStmtObjectList xEOS .
xPointerStmtObjectList:
  xPointerStmtObject / xPointerStmtObjectList ',' xPointerStmtObject .
xPointerStmtObject: xObjectName / xObjectName '(' xDeferredShapeSpecList ')' .

% R528
xTargetStmt:
  xLblDef 'target' ':' ':' xTargetObjectList xEOS /
  xLblDef 'target' xTargetObjectList xEOS .
xTargetObjectList:
  xTargetObject / xTargetObjectList ',' xTargetObject .
xTargetObject: xObjectName / xObjectName '(' xArraySpec ')' .

% R529 see 27

% R530 see 27

% R531 see 27

% R532 see 27

% R533 see 100
xConstant:
  xStructureConstructor /
```

```
  xBozLiteralConstant .

% R534 chain rule deleted

% R535 see 28

% R536 see 28
xDataIDoObject: xStructureComponent .

% R537 chain rule deleted

% R538 see 23

% R539 see 23

% R540 see 21
xImplicitStmt:
  xLblDef 'implicit' 'none' xEOS .

% R541 see 21

% R542 see 21

% R543
xNamelistStmt:
  xLblDef 'namelist' xNamelistGroups xEOS .
xNamelistGroups:
  '/' xNamelistGroupName '/' xNamelistGroupObject /
  xNamelistGroups '/' xNamelistGroupName '/' xNamelistGroupObject /
  xNamelistGroups ',' '/' xNamelistGroupName '/' xNamelistGroupObject /
  xNamelistGroups ',' xNamelistGroupObject .

% R544
xNamelistGroupObject: xVariableName .

% R545 see 17

% R546 see 17

% R547 see 18

% R548 see 19

% R549 see 19
```

```
% R601 /* ?????????????????????? */
xComplexDataRef:
  xName '%' xName /
  xComplexDataRef '%' xName .

% R603-R608 chain rules deleted

% R609 see 89

% R610 see 89

% R611 see 89

% R614
xStructureComponent:
  xVariableName xFieldSelector / xStructureComponent xFieldSelector .
xFieldSelector: '(' xSectionSubscriptList ')' '%' xName / '%' xName .

% R615
xArrayElement:
  xStructureComponent '(' xSectionSubscriptList ')' .

% R619
xSubscriptTriplet:
  xExpr ':' xExpr ':' xExpr /
  xExpr ':' ':' xExpr /
  ':' xExpr ':' xExpr /
  ':' ':' xExpr .

% R622
xAllocateStmt:
  xLblDef 'allocate' '(' xAllocationList ',' 'stat=' xVariable ')' xEOS /
  xLblDef 'allocate' '(' xAllocationList ')' xEOS .

% R623 chain rule deleted

% R624
xAllocationList: xAllocation / xAllocationList ',' xAllocation .
xAllocation: xAllocateObject / xAllocateObject xAllocatedShape .
xAllocatedShape: '(' xSectionSubscriptList ')' .
/* Need to use xSectionSubscriptList here to solve an LALR(1) conflict with the
 * xFieldSelector in R625.  (Can't tell which we have until the character
 * following the right paren, but we must reduce WITHIN the parens.)
```

```
 */

% R625
xAllocateObjectList:
  xAllocateObject / xAllocateObjectList ',' xAllocateObject .
xAllocateObject:
  xVariableName / xAllocateObject xFieldSelector .

% R626
/* Omitted to solve LALR(1) conflict.  see R624
 *
 * xAllocateShapeSpec: xExpr / xExpr ':' xExpr .
 */

% R627 chain rule deleted

% R628 chain rule deleted

% R629
xNullifyStmt:
  xLblDef 'nullify' '(' xPointerObjectList ')' xEOS .
xPointerObjectList:
  xPointerObject / xPointerObjectList ',' xPointerObject .

% R630
xPointerObject: xName / xPointerField .
xPointerField:
  xName '(' xSFExprList ')' '%' xName /
  xName '(' xSFDummyArgNameList ')' '%' xName /
  xName '%' xName /
  xPointerField xFieldSelector .

% R631
xDeallocateStmt:
  xLblDef 'deallocate' '(' xAllocateObjectList',' 'stat=' xVariable ')' xEOS /
  xLblDef 'deallocate' '(' xAllocateObjectList ')' xEOS .

% R701 see 74-85
xPrimary:
  xArrayConstructor .
xSFPrimary:
  xArrayConstructor .

% R702 chain rule deleted
```

```
% R703
xLevel1Expr:
  xDefinedUnaryOp xPrimary .

% R704
xDefinedUnaryOp:
  xDop .

% R708 see 74-81

% R709 see 74-81

% R710 see 74-81

% R712 see 82,83

% R714 see 87
xRelOp: '==' / '/=' / '<' / '<=' / '>' / '>=' .

% R719 see 84,85

% R720 see 84,85

% R721 see 84,85

% R722 see 84,85

% R723
xExpr:
  xExpr xDefinedBinaryOp xLevel5Expr .

% R724
xDefinedBinaryOp:
  xDop .

% R725-R734 chain rule deleted

% R735 see 29
xAssignmentStmt:
  xLblDef xName '%' xName '=' xExpr xEOS /
  xLblDef xName '%' xComplexDataRef '=' xExpr xEOS /
  xLblDef xName '(' xSFExprList ')' '%' xName '=' xExpr xEOS /
  xLblDef xName '(' xSFExprList ')' '%' xComplexDataRef '=' xExpr xEOS /
```

```
  xLblDef xName '(' xSFDummyArgNameList ')' '%' xName '=' xExpr xEOS /
  xLblDef xName '(' xSFDummyArgNameList ')' '%' xComplexDataRef '=' xExpr xEOS .

xSFExprList:
  xSFExpr ':' xExpr ':' xExpr /
  xSFExpr ':' ':' xExpr /
  ':' xExpr ':' xExpr /
  ':' ':' xExpr .

% R736
xPointerAssignmentStmt:
  xLblDef xName '=>' xTarget xEOS /
  xLblDef xName '%' xName '=>' xTarget xEOS /
  xLblDef xName '%' xComplexDataRef '=>' xTarget xEOS /
  xLblDef xName '(' xSFExprList ')' '%' xName '=>' xTarget xEOS /
  xLblDef xName '(' xSFExprList ')' '%' xComplexDataRef '=>' xTarget xEOS /
  xLblDef xName '(' xSFDummyArgNameList ')' '%' xName '=>' xTarget xEOS /
  xLblDef xName '(' xSFDummyArgNameList ')' '%' xComplexDataRef '=>'
    xTarget xEOS .

% R737
xTarget: xExpr .

% R738
xWhereStmt:
  xLblDef 'where' '(' xMaskExpr ')' xAssignmentStmt .

% R739
xWhereConstruct:
  xWhere xEndWhereStmt /
  xElseWhere xEndWhereStmt .
xWhere: xWhereConstructStmt / xWhere xAssignmentStmt .
xElseWhere: xWhere xElsewhereStmt / xElseWhere xAssignmentStmt .

% R740
xWhereConstructStmt:
  xLblDef 'where' '(' xMaskExpr ')' xEOS .

% R741
xMaskExpr: xExpr .

% R742
xElsewhereStmt:
  xLblDef 'elsewhere' xEOS .
```

```
% R743
xEndWhereStmt:
  xLblDef 'endwhere' xEOS /
  xLblDef 'end' 'where' xEOS .

% R801 see the note on R818

% R802 see 35

% R803 see 36

% R804 see 37
xElseIfStmt:
  xLblDef 'else' 'if' '(' xExpr ')' 'then' xEOS .

% R805 see 38

% R806 see 39
xEndIfStmt:
  xLblDef 'end' 'if' xEOS .

% R807 see 35

% R808
xCaseConstruct:
  xLblDef xName ':' 'selectcase' '(' xExpr ')' xEOS xSelectCaseRange /
  xLblDef 'selectcase' '(' xExpr ')' xEOS xSelectCaseRange /
  xLblDef xName ':' 'select' 'case' '(' xExpr ')' xEOS xSelectCaseRange /
  xLblDef 'select' 'case' '(' xExpr ')' xEOS xSelectCaseRange .
xSelectCaseRange:
  xSelectCaseBody xEndSelectStmt /
  xEndSelectStmt .

xSelectCaseBody:
  xCaseStmt / xSelectCaseBody xCaseBodyConstruct .
xCaseBodyConstruct:
  xCaseStmt / xExecutionPartConstruct .

% R810
xCaseStmt:
  xLblDef 'case' xCaseSelector xEOS /
  xLblDef 'case' xCaseSelector xName xEOS .
```

```
% R811
xEndSelectStmt:
  xLblDef 'endselect' [xEndName] xEOS /
  xLblDef 'end' 'select' [xEndName] xEOS .

% R812 chain rules deleted

% R813
xCaseSelector: '(' xCaseValueRangeList ')' / 'default' .
xCaseValueRangeList:
  xCaseValueRange / xCaseValueRangeList ',' xCaseValueRange .

% R814
xCaseValueRange:
   xExpr / xExpr ':' / ':' xExpr / xExpr ':' xExpr .

% R815 chain rules deleted

% R816 see 40
xDoConstruct: xBlockDoConstruct .

% R817
/* Block DO constructs cannot be recognized syntactically because there is
 * no requirement that there be an end do statement.
 *
 * xBlockDoConstruct:
 *   xDoStmt xBlock xEndDoStmt /
 *   xDoStmt xBlock .
 */

% R818
xBlockDoConstruct:
  xLblDef 'do' xLblRef xEOS /
  xLblDef 'do' xCommaLoopControl xEOS /
  xLblDef 'do' xEOS /
  xLblDef xName ':' 'do' xLblRef xCommaLoopControl xEOS /
  xLblDef xName ':' 'do' xLblRef xEOS /
  xLblDef xName ':' 'do' xCommaLoopControl xEOS /
  xLblDef xName ':' 'do' xEOS .

% R819 chain rule deleted

% R820 chain rule deleted
```

```
% R821 see 40
xLoopControl: 'while' '(' xExpr ')' .

% R822 chain rule deleted

% R823 chain rule deleted

% R824 see note on R818

% R825
xEndDoStmt:
  xLblDef 'enddo' [xEndName] xEOS /
  xLblDef 'end' 'do' [xEndName] xEOS .

% R826-R833 enforced semantically

% R834
xCycleStmt:
  xLblDef 'cycle' [xEndName] xEOS .

% R835
xExitStmt:
  xLblDef 'exit' [xEndName] xEOS .

% R836 see 31
GoToKw: 'go' 'to' .

% R837 see 32

% R838 see 29

% R839 see 33

% R840 see 34

% R841 see 41

% R842 see 42

% R843 see 42

% R901 see 56

% R902 chain rule deleted
```

```
% R903 chain rule deleted

% R904 see 50

% R905 see 50
xConnectSpec:
  'position=' xCExpr /
  'action=' xCExpr /
  'delim=' xCExpr /
  'pad=' xCExpr .

% R906 chain rule deleted

% R907 see 51

% R908 see 51

% R909 see 45

% R910 see 44

% R911 see 46

% R912 see 46
xIoControlSpec:
  'nml=' xNamelistGroupName /
  'advance=' xCExpr /
  'size=' xVariable /
  'eor=' xLblRef .

% R913 see 57

% R920 see 54
xEndfileStmt:
  xLblDef 'end' 'file' xUnitIdentifier xEOS /
  xLblDef 'end' 'file' '(' xPositionSpecList ')' xEOS .

% R923 see 52
xInquireStmt:
  xLblDef 'inquire' '(' 'iolength=' xScalarVariable ')' xOutputItemList xEOS .

% R924 see 52
xInquireSpec:
```

```
  'position=' xScalarVariable /
  'action=' xScalarVariable /
  'read=' xScalarVariable /
  'write=' xScalarVariable /
  'readwrite=' xScalarVariable /
  'delim=' xScalarVariable /
  'pad=' xScalarVariable .

% R1101 see 2
xMainRange: xBodyPlusInternals xEndProgramStmt .

% R1102 see 8

% R1103
xEndProgramStmt:
  xLblDef 'endprogram' [xEndName] xEOS /
  xLblDef 'end' 'program' [xEndName] xEOS .

% R1104
xModule:
  xModuleStmt xModuleBody xEndModuleStmt /
  xModuleStmt xEndModuleStmt .
xModuleBody:
  xSpecificationPartConstruct /
  xModuleSubprogramPartConstruct /
  xModuleBody xSpecificationPartConstruct /
  xModuleBody xModuleSubprogramPartConstruct .

% R1105
xModuleStmt:
  xLblDef 'module' xModuleName xEOS .

% R1106
xEndModuleStmt:
  xLblDef 'endmodule' [xEndName] xEOS /
  xLblDef 'end' 'module' [xEndName] xEOS /
  xLblDef 'end' xEOS .

% R1107
xUseStmt:
  xLblDef 'use' xName xEOS /
  xLblDef 'use' xName ',' 'only' ':' xEOS /
  xLblDef 'use' xName ',' xRenameList xEOS /
  xLblDef 'use' xName ',' 'only' ':' xOnlyList xEOS .
```

```
xRenameList: xRename / xRenameList ',' xRename .
xOnlyList: xOnly / xOnlyList ',' xOnly .

% R1108
xRename: xIdent '=>' xUseName .

% R1109
xOnly: xGenericSpec / xIdent '=>' xUseName / xUseName .

% R1110 see 5

% R1111 see 14

% R1112
xEndBlockDataStmt:
  xLblDef 'endblockdata' [xEndName] xEOS /
  xLblDef 'end' 'blockdata' [xEndName] xEOS /
  xLblDef 'endblock' 'data' [xEndName] xEOS /
  xLblDef 'end' 'block' 'data' [xEndName] xEOS .

% R1201
xInterfaceBlock:
  xInterfaceStmt xInterfaceBlockBody xEndInterfaceStmt .
xInterfaceBlockBody:
  xInterfaceBodyPartConstruct /
  xInterfaceBlockBody xInterfaceBodyPartConstruct .
xInterfaceBodyPartConstruct: xInterfaceBody / xModuleProcedureStmt .

% R1202
xInterfaceStmt:
  'interface' xGenericName xEOS /
  'interface' xGenericSpec xEOS /
  'interface' xEOS .

% R1203
xEndInterfaceStmt:
  xLblDef 'endinterface' xEOS /
  xLblDef 'end' 'interface' xEOS .

% R1204
xInterfaceBody:
  xLblDef xFunctionPrefix xFunctionName xFunctionInterfaceRange /
  xLblDef 'subroutine' xSubroutineName xSubroutineInterfaceRange .
xFunctionInterfaceRange:
```

```
  xFunctionParList xEOS xSubprogramInterfaceBody xEndFunctionStmt /
  xFunctionParList xEOS xEndFunctionStmt .
xSubroutineInterfaceRange:
  xSubroutineParList xEOS xSubprogramInterfaceBody xEndSubroutineStmt /
  xSubroutineParList xEOS xEndSubroutineStmt .
xSubprogramInterfaceBody:
  xSpecificationPartConstruct /
  xSubprogramInterfaceBody xSpecificationPartConstruct .

% R1205
xModuleProcedureStmt:
  xLblDef 'module' 'procedure' xProcedureNameList xEOS .
xProcedureNameList:
  xProcedureName / xProcedureNameList ',' xProcedureName .
xProcedureName: xIdent .

% R1206
xGenericSpec:
  'operator' '(' xDefinedOperator ')' /
  'assignment' '(' '=' ')' .

% R1207 see 24

% R1208 see 25

% R1209 see 73
xFunctionReference: xName '(' xFunctionArgList ')' .

% R1210 see 71

% R1211
xFunctionArgList:
  xFunctionArg /
  xFunctionArgList ',' xFunctionArg /
  xSectionSubscriptList ',' xFunctionArg .
xFunctionArg: xName '=' xExpr .
xSubroutineArg: xName '=' xExpr / xName '=' xHcon / xName '=' '*' xLblRef .

% R1212 chain rule deleted

% R1213 chain rule deleted

% R1214 chain rule deleted
```

```
% R1215 see 3
xFunctionRange:
  xFunctionParList 'result' '(' xName ')' xEOS xBodyPlusInternals
    xEndFunctionStmt /
  xFunctionParList 'result' '(' xName ')' xEOS xBody xEndFunctionStmt /
  xFunctionParList 'result' '(' xName ')' xEOS xEndFunctionStmt /
  xFunctionParList xEOS xBodyPlusInternals xEndFunctionStmt .

% R1216 see 10

% R1217 see 10
xFunctionPrefix:
  'recursive' 'function' /
  'recursive' xTypeSpec 'function' /
  xTypeSpec 'recursive' 'function' .

% R1218
xEndFunctionStmt:
  xLblDef 'endfunction' [xEndName] xEOS /
  xLblDef 'end' 'function' [xEndName] xEOS .

% R1219 see 4
xSubroutineSubprogram:
  xLblDef 'recursive' 'subroutine' xSubroutineName xSubroutineRange .
xSubroutineRange:
  xSubroutineParList xEOS xBodyPlusInternals xEndSubroutineStmt .

% R1220 see 12

% R1221 see 12

% R1222
xEndSubroutineStmt:
  xLblDef 'endsubroutine' [xEndName] xEOS /
  xLblDef 'end' 'subroutine' [xEndName] xEOS .

% R1223 see 13
xEntryStmt:
  xLblDef 'entry' xEntryName xSubroutineParList 'result' '(' xName ')' xEOS .

% R1224 see 72

% R1225
xContainsStmt:
```

```
  xLblDef 'contains' xEOS.

% R1226 see 70

/* Nonterminal symbols that are not defined in the standard
 */

xArrayName: xIdent .
xComponentName: xIdent .
xGenericName: xIdent .
xModuleName: xIdent .
xNamelistGroupName: xIdent .
xUseName: xIdent .
xTypeName: xIdent .
xEndName: xIdent .
```

This macro is invoked in definition 18.

2 Changes Due to LALR(1) Conflicts

2.1 Entry Statement

Entry Statement[4] ≡

```
xEntryStmt:
  xLblDef 'entry' xEntryName xSubroutineParList xEOS .
```

This macro is invoked in definition 2.

2.2 Specification Expression

Specification Expression[5] ≡
`xExpr`This macro is invoked in definition 2.

2.3 Equivalence Entity

Equivalence Entity[6] ≡
`xVariable`This macro is invoked in definition 2.

2.4 Letter Specifications

Letter Specifications[7] ≡
    `xImpl`This macro is invoked in definition 2.

2.5 Scalar Integer Literal Constant

Scalar Integer Literal Constant[8] ≡
    `xIcon`This macro is invoked in definition 2.

2.6 Constant Expression

Constant Expression[9] ≡
    `xExpr`This macro is invoked in definition 2.

2.7 Integer Expression

Integer Expression[10] ≡
    `xExpr`This macro is invoked in definition 2.

2.8 Int-Real-Dp Expression

Int-Real-Dp Expression[11] ≡
    `xExpr`This macro is invoked in definition 2.

2.9 Scalar Logical Expression

Scalar Logical Expression[12] ≡
    `xExpr`This macro is invoked in definition 2.

2.10 Assignment Statement

Assignment Statement[13] ≡

```
xAssignmentStmt:
  xLblDef xName '=' xExpr xEOS /
  xLblDef xName '(' xSFExprList ')' '=' xExpr xEOS /
  xLblDef xName '(' xSFExprList ')' xSubstringRange '=' xExpr xEOS .

xSFExprList:
  ':' /
  ':' xExpr /
  xSFExpr /
  xSFExpr ':' /
  xSFExpr ':' xExpr /
  xSFExprList ',' xSectionSubscript /
  xSFDummyArgNameList ',' ':' /
  xSFDummyArgNameList ',' ':' xExpr /
  xSFDummyArgNameList ',' xSFExpr /
  xSFDummyArgNameList ',' xSFExpr ':' /
  xSFDummyArgNameList ',' xSFExpr ':' xExpr .
xSFExpr: xSFTerm / xSign xAddOperand / xSFExpr xAddOp xAddOperand .
xSFTerm: xSFFactor / xSFTerm xMultOp xMultOperand .
xSFFactor: xSFPrimary / xSFPrimary xPowerOp xMultOperand .
xSFPrimary:
  xIcon /
  xSFVarName /
  xComplexDataRef /
  xFunctionReference /
  '(' xExpr ')' .


xAssignStmt:
  xLblDef 'assign' xLblRef 'to' xVariableName xEOS .
```

This macro is invoked in definition 2.

3 Semantically-Equivalent Grammar Symbols

Many of the symbols used in the context-free grammar represent semantically-equivalent phrases. Some are used to make the prose of the standard clearer, others avoid ambiguities. There is no reason to distinguish them once the abstract syntax tree is built, and so we replace all of the symbols in an equivalence class by a single member of that class.

3.1 Equivalence Classes Common to FORTRAN 77 and FORTRAN 90

Equivalence Classes Common to FORTRAN 77 and FORTRAN 90[14] ≡

```
xProgramUnit ::=
       xMainProgram xSubroutineSubprogram xFunctionSubprogram
       xBlockDataSubprogram.

xSubprogramRange ::=
       xFunctionRange xSubroutineRange .

xComblock ::=
       xSavedCommonBlock .

xFormalParameter ::=
       xFunctionPar xSubroutinePar.

xFormatIdentifier ::=
       xRdFmtId.

xFormalParameterList ::=
       xFunctionParList xFunctionPars xSubroutineParList xSubroutinePars.

xBody ::=
       xConditionalBody xBlockDataBody .

xBodyConstruct ::=
       xSpecificationPartConstruct xDeclarationConstruct
       xExecutionPartConstruct xExecutableConstruct
       xBlockDataBodyConstruct .

xLoopControl ::=
       xCommaLoopControl.

xCharSelector ::=
       xLengthSelector.

xArg ::=
       xSubroutineArg .

xArgList ::=
       xSubroutineArgList .
```

```
xExpr ::=
       xPrimary xLevel1Expr xLevel2Expr xLevel3Expr xLevel4Expr xLevel5Expr
       xFunctionReference
       xAddOperand xMultOperand xEquivOperand xOrOperand xAndOperand
       xSFPrimary xSFTerm xSFFactor xSFExpr
       xUFPrimary xUFExpr xUFTerm xUFFactor
       xCPrimary xCExpr xCOperand
       xRdFmtIdExpr xCommaExp .

xExprList ::=
       xSFExprList .

xStmt ::=
       xSpecificationStmt xActionStmt
       xImplicitStmt
       xIfThenStmt xElseIfStmt xElseStmt
       xParameterStmt xFormatStmt xEntryStmt xTypeDeclarationStmt
       xCommonStmt xDataStmt xDimensionStmt
       xEquivalenceStmt xExternalStmt xIntrinsicStmt xSaveStmt
       xArithmeticIfStmt
       xAssignmentStmt xAssignStmt xBackspaceStmt xCallStmt xCloseStmt
       xContinueStmt xEndfileStmt xGotoStmt xComputedGotoStmt
       xAssignedGotoStmt xIfStmt xInquireStmt xOpenStmt xPauseStmt
       xPrintStmt xReadStmt xReturnStmt xRewindStmt xStmtFunctionStmt
       xStopStmt xWriteStmt xLabelDoStmt .

xBinOp ::=
       xConcatOp xPowerOp xMultOp xAddOp xRelOp xAndOp xOrOp xEquivOp.

xUnOp ::=
       xNotOp xSign .

xVariableName ::=
       xVariableComma .

xIoControlSpec ::=
       xCloseSpec xRdCtlSpec xConnectSpec xInquireSpec xPositionSpec .

xIoControlSpecList ::=
       xCloseSpecList xRdIoCtlSpecList xPositionSpecList xConnectSpecList
       xInquireSpecList .

xArraySpec ::=
       xExplicitShapeSpecList xAssumedSizeSpec.
```

```
xDataStmtObject ::=
       xDataIDoObject.

xOutputItemList ::=
       xOutputItemList1 .

xFmtSpec ::=
       xFormatedit xFormatsep xEditElement.
```

This macro is invoked in definition 20.

3.2 Equivalence Classes Specific to FORTRAN 90

Equivalence Classes Specific to FORTRAN 90[15] ≡

```
xProgramUnit ::=
       xModule xModuleSubprogram .

xHeader ::=
  xModuleStmt
  .

xBody ::=
       xBlockDoConstruct xModuleBody xSubprogramInterfaceBody
       xInterfaceBody xInterfaceBlockBody xCaseConstruct
       xSelectCaseBody xWhere xWhereConstruct xElseWhere
       xBodyPlusInternals .

xLine ::=
  xEndModuleStmt
  .

xSubprogramRange ::=
  xFunctionInterfaceRange xSubroutineInterfaceRange .

xArg ::=
       xFunctionArg .

xArgList ::=
       xFunctionArgList .
```

```
xExpr ::=
       xArrayConstructor xTarget xMaskExpr .

xPointerObject ::=
       xPointerField.

xAcValueList ::=
       xAcValueList1.

xStmt ::=
       xUseStmt
       xDerivedTypeDef xModuleSubprogramPartConstruct xContainsStmt
       xAccessStmt xAllocatableStmt
       xIntentStmt xNamelistStmt xOptionalStmt xPointerStmt xTargetStmt
       xEndDoStmt
       xAllocateStmt xCycleStmt xDeallocateStmt xExitStmt
       xNullifyStmt xPointerAssignmentStmt
       xWhereStmt xElsewhereStmt xWhereConstructStmt
       xCaseStmt xCaseBodyConstruct
       xDerivedTypeBodyConstruct
       xPrivateSequenceStmt xComponentDefStmt
       xInterfaceBodyPartConstruct xModuleProcedureStmt.

xArraySpec ::=
       xComponentArraySpec xExplicitShapeSpecList xDeferredShapeSpecList
       xAssumedShapeSpecList xAssumedSizeSpec.
```

This macro is invoked in definition 20.

4 Literal Terminals Recognized by special means

Under normal circumstances, Eli uses a finite-state machine to recognize all of the literal terminal symbols of a context-free grammar. FORTRAN keywords and operators like `.EQ.`, however, are recognized specially by the scanner. Thus they must *not* be included in the finite-state machine.

We prevent them from being included by providing a specification to recognize their appearance in the grammar. This specification is written using GLA regular expressions:

Literal Terminals Recognized by special means[16] ≡

```
Keyword:        $[a-zA-Z][a-zA-Z]*=?
xDefop: $\.[a-zA-Z]*\.
```

This macro is invoked in definition 19.

`Keyword` recognizes keywords, including I/O controls followed by `=`; `xDefop` recognizes operators delimited by dots.

The specification is output as a file not belonging directly to the specification set, but information is derived from it:

Derivation of a terminal symbol extractor[17] ≡

```
grammar.gla :kwd
```

This macro is invoked in definition 21.

5 Output files

5.1 grammar.con

A type-`con` file contains the context-free grammar describing the way a program is written.

grammar.con[18] ≡

```
xSourceFile : xExecutableProgram.

Productions Common to FORTRAN 77 and FORTRAN 90[2]
#if !Fortran77[1]
Productions Specific to FORTRAN 90[3]
#endif
```

This macro is attached to a product file.

5.2 grammar.gla

A type-`gla` file contains regular expressions defining a set of strings. The `@N` operation marks this file as a "non-product file"; a file that is used in the derivation of product components, but is not itself a component of the product.

grammar.gla[19] ≡

Literal Terminals Recognized by special means[16]

This macro is attached to a non-product file.

5.3 grammar.map

A type-`map` file contains the equivalence classes for the abstract syntax.

grammar.map[20] ≡

```
MAPSYM
```
Equivalence Classes Common to FORTRAN 77 and FORTRAN 90[14]
`#if !`*Fortran77*[1]
Equivalence Classes Specific to FORTRAN 90[15]
`#endif`

This macro is attached to a product file.

5.4 grammar.specs

A type-`specs` file contains derivations for components of the product.

grammar.specs[21] ≡

Derivation of a terminal symbol extractor[17]

This macro is attached to a product file.