Albert Ludwigs Universität Freiburg

Technische Fakultät

PicoC-Compiler

Übersetzung einer Untermenge von C in den Befehlssatz der RETI-CPU

Bachelorarbeit

Abgabedatum: 28th April 2022

Author:
Jürgen Mattheis

Gutachter:
Prof. Dr. Scholl

Betreung:
M.Sc. Seufert

Eine Bachelorarbeit am Lehrstuhl für
Betriebssysteme

ERKLÄRUNG

Hiermit erkläre ich, dass ich diese Abschlussarbeit selbständig verfasst habe, keine anderen als die angegebenen Quellen/Hilfsmittel verwendet habe und alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten Schriften entnommen wurden, als solche kenntlich gemacht habe. Darüber hinaus erkläre ich, dass diese Abschlussarbeit nicht, auch nicht auszugsweise, bereits für eine andere Prüfung angefertigt wurde.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Definitionen

1 Implementierung

1.1 Architektur

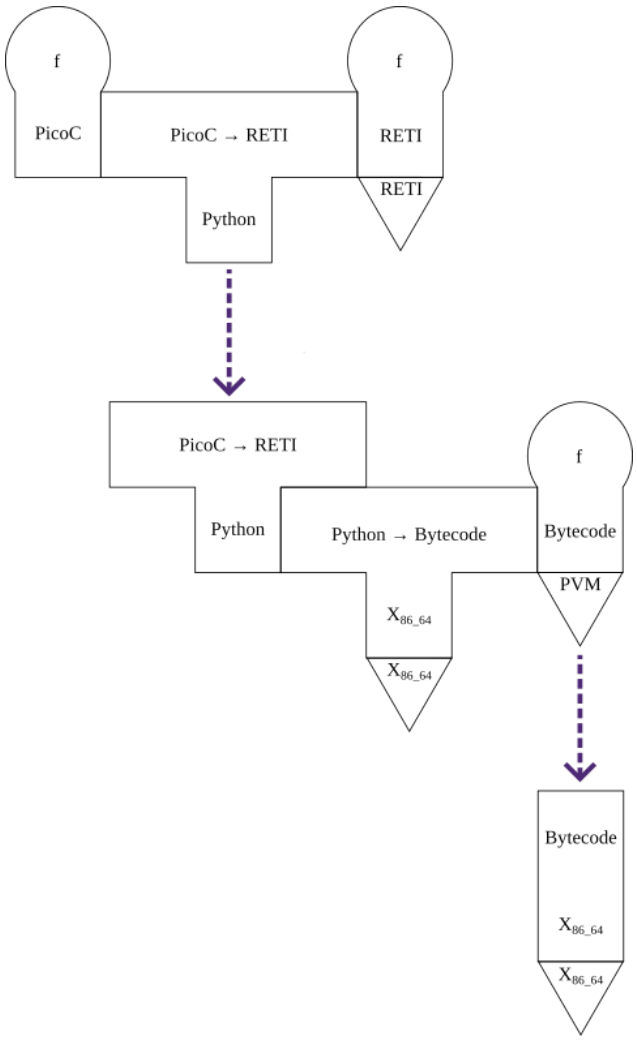

Abbildung 1.1: Cross-Compiler Kompiliervorgang ausgeschrieben

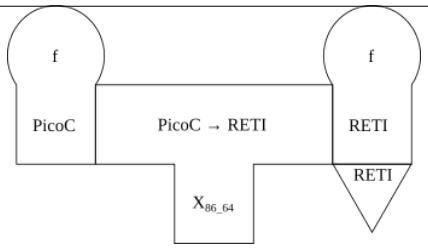

Abbildung 1.2: Cross-Compiler Kompiliervorgang Kurzform

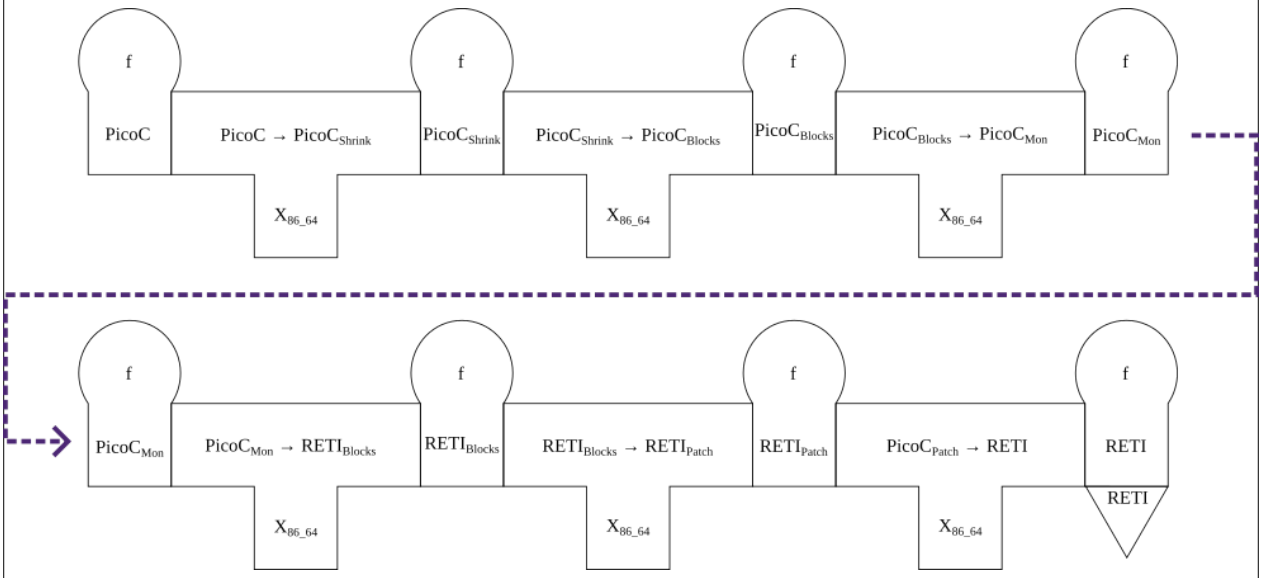

Abbildung 1.3: Architektur mit allen Passes ausgeschrieben

1.2 Lexikalische Analyse

1.2.1 Verwendung von Lark

1.2.2 Basic Parser

1.3 Syntaktische Analyse

1.3.1 Verwendung von Lark

1.3.2 Umsetzung von Präzidenz

Die **PicoC Sprache** hat dieselben **Präzidenzregeln** implementiert, wie die **Sprache C**[1]. Die **Präzidenzregeln** von **PicoC** sind in Tabelle 1.1 aufgelistet.

[1] *C Operator Precedence - cppreference.com.*

Präzidenz	Operator	Beschreibung	Assoziativität

1	`a()`
	`a[]`
	`a.b`
		Funktionsaufruf
		Indexzugriff
		Attributzugriff
			Links, dann rechts →

2	`-a`
	`!a ~a`
	`*a &a`
		Unäres Minus
		Logisches NOT und Bitweise NOT
		Dereferenz und Referenz, auch Adresse-von
			Rechts, dann links ←

3	`a*b a/b a%b`	Multiplikation, Division und Modulo	Links, dann rechts →
4	`a+b a-b`	Addition und Subtraktion	
5	`a<b a<=b` `a>b a>=b`	Kleiner, Kleiner Gleich, Größer, Größer gleich	
6	`a==b a!=b`	Gleichheit und Ungleichheit	
7	`a&b`	Bitweise UND	
8	`a^b`	Bitweise XOR (exclusive or)	
9	`a|b`	Bitweise ODER (inclusive or)	
10	`a&&b`	Logiches UND	
11	$a \| \| b$	Logisches ODER	
12	`a=b`	Zuweisung	Rechts, dann links ←
13	`a,b`	Komma	Links, dann rechts →

Tabelle 1.1: Präzidenzregeln von PicoC

1.3.3 Derivation Tree Generierung

1.3.4 Early Parser

1.3.5 Derivation Tree Vereinfachung

1.3.6 Abstrakt Syntax Tree Generierung

1.3.6.1 ASTNode

1.3.6.2 PicoC Nodes

1.3.6.3 RETI Nodes

1.4 Code Generierung

1.4.1 Passes

1.4.1.1 PicoC-Shrink Pass

1.4.1.2 PicoC-Blocks Pass

1.4.1.3 PicoC-Mon Pass

Definition 1.1: Symboltabelle

1.4.1.4 RETI-Blocks Pass

1.4.1.5 RETI-Patch Pass

1.4.1.6 RETI Pass

1.4.2 Umsetzung von Pointern

1.4.2.1 Pointer Definition

1.4.2.2 Pointer Dereferenzierung durch Zugriff auf Arrayindex ersetzen

```
void main() {
  int var = 42;
  int *pntr = &var;
  *pntr;
}
```

Abbildung 1.4: PicoC Code für Pointer Dereferenzierung

```
File
  Name './code_examples/example_pntr_deref.ast',
  [
    FunDef
      VoidType 'void',
      Name 'main',
      [],
      [
        Assign
          Alloc
            Writeable,
            IntType 'int',
            Name 'var',
          Num '42',
        Assign
          Alloc
            Writeable,
            PntrDecl
              Num '1',
              IntType 'int',
            Name 'pntr',
          Ref
            Name 'var',
        Exp
          Deref
            Name 'pntr',
            Num '0'
      ]
  ]
```

Abbildung 1.5: Abstract Syntax Tree für Pointer Dereferenzierung

```
File
  Name './code_examples/example_pntr_deref.picoc_shrink',
  [
    FunDef
      VoidType 'void',
      Name 'main',
      [],
      [
        Assign
          Alloc
            Writeable,
            IntType 'int',
            Name 'var',
          Num '42',
        Assign
          Alloc
            Writeable,
            PntrDecl
              Num '1',
              IntType 'int',
            Name 'pntr',
          Ref
            Name 'var',
        Exp
          Subscr
            Name 'pntr',
            Num '0'
      ]
  ]
```

Abbildung 1.6: PicoC Shrink Pass für Pointer Dereferenzierung

1.4.2.3 Referenzierung

1.4.3 Umsetzung von Arrays

1.4.3.1 Definition von Arrays

1.4.3.2 Initialisierung von Arrays

```
void main() {
  int ar[2][1] = {{4}, {2}};
}
```

Abbildung 1.7: PicoC Code für Array Initialisierung

```
File
  Name './code_examples/example_array_init.ast',
  [
    FunDef
      VoidType 'void',
      Name 'main',
      [],
      [
        Assign
          Alloc
            Writeable,
            ArrayDecl
              [
                Num '2',
                Num '1'
              ],
              IntType 'int',
            Name 'ar',
          Array
            [
              Array
                [
                  Num '4'
                ],
              Array
                [
                  Num '2'
                ]
            ]
      ]
  ]
```

Abbildung 1.8: Abstract Syntax Tree für Array Initialisierung

```
File
  Name './code_examples/example_array_init.picoc_mon',
  [
    Block
      Name 'main.0',
      [
        Exp
          Num '4',
        Exp
          Num '2',
        Assign
          GlobalWrite
            Num '0',
          Tmp
            Num '2',
        Return
          Empty
      ]
  ]
```

Abbildung 1.9: PicoC Mon Pass für Array Initialisierung

```
File
  Name './code_examples/example_array_init.reti_blocks',
  [
    Block
      Name 'main.0',
      [
        SUBI SP 1,
        LOADI ACC 4,
        STOREIN SP ACC 1,
        SUBI SP 1,
        LOADI ACC 2,
        STOREIN SP ACC 1,
        LOADIN SP ACC 1,
        STOREIN DS ACC 1,
        LOADIN SP ACC 2,
        STOREIN DS ACC 0,
        ADDI SP 2,
        LOADIN BAF PC -1
      ]
  ]
```

Abbildung 1.10: RETI Blocks Pass für Array Initialisierung

1.4.3.3 Zugriff auf Arrayindex

Der Zugriff auf einen bestimmten Index eines Arrays ist wie folgt umgesetzt:

```
void main() {
  int ar[2] = {1, 2};
  ar[2];
}
```

Abbildung 1.11: PicoC Code für Zugriff auf Arrayindex

```
File
  Name './example_array_access.ast',
  [
    FunDef
      VoidType 'void',
      Name 'main',
      [],
      [
        Assign
          Alloc
            Writeable,
            ArrayDecl
              [
                Num '2'
              ],
              IntType 'int',
            Name 'ar',
          Array
            [
              Num '1',
              Num '2'
            ],
        Exp
          Subscr
            Name 'ar',
            Num '2'
      ]
  ]
```

Abbildung 1.12: Abstract Syntax Tree für Zugriff auf Arrayindex

```
File
  Name './example_array_access.picoc_mon',
  [
    Block
      Name 'main.0',
      [
        Exp
          Num '1',
        Exp
          Num '2',
        Assign
          GlobalWrite
            Num '0',
          Tmp
            Num '2',
        Ref
          GlobalRead
            Num '0',
        Exp
          Num '2',
        Ref
          Subscr
            Tmp
              Num '2',
            Tmp
              Num '1',
        Exp
          Subscr
            Tmp
              Num '1',
            Num '0',
        Return
          Empty
      ]
  ]
```

Abbildung 1.13: PicoC Mon Pass für Zugriff auf Arrayindex

```
File
  Name './example_array_access.reti_blocks',
  [
    Block
      Name 'main.0',
      [
        SUBI SP 1,
        LOADI ACC 1,
        STOREIN SP ACC 1,
        SUBI SP 1,
        LOADI ACC 2,
        STOREIN SP ACC 1,
        LOADIN SP ACC 1,
        STOREIN DS ACC 1,
        LOADIN SP ACC 2,
        STOREIN DS ACC 0,
        ADDI SP 2,
        SUBI SP 1,
        LOADI IN1 0,
        ADD IN1 DS,
        STOREIN SP IN1 1,
        SUBI SP 1,
        LOADI ACC 2,
        STOREIN SP ACC 1,
        LOADIN SP IN1 2,
        LOADIN SP IN2 1,
        MULTI IN2 1,
        ADD IN1 IN2,
        ADDI SP 1,
        STOREIN SP IN1 1,
        LOADIN SP IN1 1,
        LOADIN IN1 ACC 0,
        STOREIN SP ACC 1,
        LOADIN BAF PC -1
      ]
  ]
```

Abbildung 1.14: RETI Blocks Pass für Zugriff auf Arrayindex

1.4.3.4 Zuweisung an Arrayindex

```
void main() {
  int ar[2];
  ar[2] = 42;
}
```

Abbildung 1.15: PicoC Code für Zuweisung an Arrayindex

```
File
  Name './example_array_assignment.ast',
  [
    FunDef
      VoidType 'void',
      Name 'main',
      [],
      [
        Exp
          Alloc
            Writeable,
            ArrayDecl
              [
                Num '2'
              ],
              IntType 'int',
            Name 'ar',
        Assign
          Subscr
            Name 'ar',
            Num '2',
          Num '42'
      ]
  ]
```

Abbildung 1.16: Abstract Syntax Tree für Zuweisung an Arrayindex

```
File
  Name './example_array_assignment.picoc_mon',
  [
    Block
      Name 'main.0',
      [
        Exp
          Num '42',
        Ref
          GlobalRead
            Num '0',
        Exp
          Num '2',
        Ref
          Subscr
            Tmp
              Num '2',
            Tmp
              Num '1',
        Assign
          Subscr
            Tmp
              Num '1',
            Num '0',
          Tmp
            Num '2',
        Return
          Empty
      ]
  ]
```

Abbildung 1.17: PicoC Mon Pass für Zuweisung an Arrayindex

```
File
  Name './example_array_assignment.reti_blocks',
  [
    Block
      Name 'main.0',
      [
        SUBI SP 1,
        LOADI ACC 42,
        STOREIN SP ACC 1,
        SUBI SP 1,
        LOADI IN1 0,
        ADD IN1 DS,
        STOREIN SP IN1 1,
        SUBI SP 1,
        LOADI ACC 2,
        STOREIN SP ACC 1,
        LOADIN SP IN1 2,
        LOADIN SP IN2 1,
        MULTI IN2 1,
        ADD IN1 IN2,
        ADDI SP 1,
        STOREIN SP IN1 1,
        LOADIN SP IN1 1,
        LOADIN SP ACC 2,
        ADDI SP 2,
        STOREIN IN1 ACC 0,
        LOADIN BAF PC -1
      ]
  ]
```

Abbildung 1.18: RETI Blocks Pass für Zuweisung an Arrayindex

1.4.4 Umsetzung von Structs

1.4.4.1 Deklaration von Structs

1.4.4.2 Definition von Structs

1.4.4.3 Initialisierung von Structs

1.4.4.4 Zugriff auf Structattribut

1.4.4.5 Zuweisung an Structattribut

1.4.5 Umsetzung des Zusammenspiels der Derived Datatypes

1.4.6 Umsetzung von Funktionen

1.4.6.1 Funktionen auflösen zu RETI Code

1.4.6.1.1 Sprung zur Main Funktion

1.4.6.2 Funktionsdeklaration

1.4.6.3 Funktionsdefinition

1.4.6.3.1 Allocation von Variablen

1.4.6.4 Funktionsaufruf

1.4.6.4.1 Mit Rückgabewert

```
void stack_fun() {
  return 42;
}

void main() {
  int var = stack_fun();
}
```

Abbildung 1.19: PicoC Code für Funktionsaufruf

1.4.6.4.2 Ohne Rückgabewert

1.4.6.4.3 Umsetzung von Call by Sharing für Arrays

1.4.6.4.4 Umsetzung von Call by Value für Structs

1.4.7 Umsetzung kleinerer Details

1.5 Fehlermeldungen

1.5.1 Error Handler

1.5.2 Arten von Fehlermeldungen

1.5.2.1 Syntaxfehler

1.5.2.2 Laufzeitfehler

A Appendix

A.1 Konkrette und Abstrakte Syntax

A.2 Bedienungsanleitungen

A.2.1 PicoC-Compiler

A.2.2 Showmode

A.2.3 Entwicklertools

Literatur

Online

- *C Operator Precedence - cppreference.com*. URL: https://en.cppreference.com/w/c/language/operator_precedence (besucht am 27. 04. 2022).