

$\langle Zi \rangle$	$\rightarrow 0 \dots 9$	Ziffer
$\langle ZiF \rangle$	$\rightarrow \langle Zi \rangle \langle ZiF \rangle \langle Zi \rangle$	Ziffernfolge
$\langle Bu \rangle$	$\rightarrow a \dots z A \dots Z$	Buchstabe
$\langle BuZi \rangle$	$\rightarrow \langle Bu \rangle \langle Zi \rangle$	alphanumerisches Zeichen
$\langle BuZiF \rangle$	$\rightarrow \langle BuZi \rangle \langle BuZi \rangle \langle BuZiF \rangle$	alphanumerische Zeichenfolge
$\langle Na \rangle$	$\rightarrow \langle Bu \rangle \langle Bu \rangle \langle BuZiF \rangle$	Name
$\langle C \rangle$	$\rightarrow \langle ZiF \rangle \langle ZiF \rangle u \text{null}$	<i>int</i> -/ <i>uint</i> -Konstante, Nullpointerkonstante
$\langle CC \rangle$	$\rightarrow '\langle BuZi \rangle'$	<i>char</i> -Konstante
$\langle BC \rangle$	$\rightarrow \text{true} \text{false}$	<i>bool</i> -Konstanten
$\langle id \rangle$	$\rightarrow \langle Na \rangle \langle id \rangle . \langle Na \rangle \langle id \rangle [\langle A \rangle] \langle id \rangle * \langle id \rangle \&$	Identifizier ((Sub-)Variablenbezeichner)
$\langle F \rangle$	$\rightarrow \langle id \rangle -_1 \langle F \rangle (\langle A \rangle) \langle C \rangle$	Faktor
$\langle T \rangle$	$\rightarrow \langle F \rangle \langle T \rangle \cdot \langle F \rangle \langle T \rangle / \langle F \rangle$	Term
$\langle A \rangle$	$\rightarrow \langle T \rangle \langle A \rangle + \langle T \rangle \langle A \rangle -_2 \langle T \rangle$	(algebraischer) Ausdruck
$\langle Atom \rangle$	$\rightarrow \langle A \rangle > \langle A \rangle \langle A \rangle \geq \langle A \rangle \langle A \rangle < \langle A \rangle \langle A \rangle \leq \langle A \rangle \langle A \rangle == \langle A \rangle \langle A \rangle \neq \langle A \rangle \langle BC \rangle$	"Boolesche Variable"
$\langle BF \rangle$	$\rightarrow \langle id \rangle \langle Atom \rangle \sim \langle BF \rangle (\langle BA \rangle)$	Boolescher Faktor
$\langle BT \rangle$	$\rightarrow \langle BF \rangle \langle BT \rangle \wedge \langle BF \rangle$	Boolescher Term
$\langle BA \rangle$	$\rightarrow \langle BT \rangle \langle BA \rangle \vee \langle BT \rangle$	Boolescher Ausdruck
$\langle An \rangle$	$\rightarrow \langle id \rangle = \langle A \rangle \langle id \rangle = \langle BA \rangle \langle id \rangle = \langle CC \rangle $ $if \langle BA \rangle \{ \langle AnF \rangle \} else \{ \langle AnF \rangle \} $ $if \langle BA \rangle \{ \langle AnF \rangle \} $ $while \langle BA \rangle \{ \langle AnF \rangle \} $ $\langle id \rangle = \langle Na \rangle (\langle PaF \rangle) $ $\langle id \rangle = new \langle Na \rangle *$	Zuweisung bedingte Anweisung Schleife Funktionsaufruf mit Parametern
$\langle rAn \rangle$	$\rightarrow return \langle A \rangle return \langle BA \rangle$	Speicher allozieren Return-Anweisung
$\langle PaF \rangle$	$\rightarrow \varepsilon \langle ParF \rangle$	Parameterfolge
$\langle ParF \rangle$	$\rightarrow \langle BA \rangle \langle A \rangle \langle A \rangle, \langle ParF \rangle \langle BA \rangle, \langle ParF \rangle$	nicht-leere Parameterfolge
$\langle AnF \rangle$	$\rightarrow \langle An \rangle \langle An \rangle ; \langle AnF \rangle$	Anweisungsfolge
$\langle program \rangle$	$\rightarrow \langle TypDF \rangle ; \langle VarDF \rangle ; \langle FunDF \rangle $ $\langle VarDF \rangle ; \langle FunDF \rangle $ $\langle TypDF \rangle ; \langle FunDF \rangle $ $\langle FunDF \rangle$	C0-Programm ohne Typdeklarationen ohne Variablendeklarationen ohne Typ- und Variablendeklarationen
$\langle TypDF \rangle$	$\rightarrow \langle TypD \rangle \langle TypD \rangle ; \langle TypDF \rangle$	Typdeklarationsfolge, nicht-leer
$\langle TypD \rangle$	$\rightarrow typedef \langle Typ \rangle \langle Na \rangle$	Typdeklaration
$\langle Typ \rangle$	$\rightarrow \langle Na \rangle [\langle ZiF \rangle] \langle Na \rangle * struct \{ \langle VarDF \rangle \}$	Array-Typ, Pointer, struct-Typ
$\langle VarDF \rangle$	$\rightarrow \langle VarD \rangle \langle VarD \rangle ; \langle VarDF \rangle$	Variablendeklarationsfolge, nicht-leer
$\langle VarD \rangle$	$\rightarrow \langle Na \rangle \langle Na \rangle$	Variablendeklaration
$\langle FunDF \rangle$	$\rightarrow \langle FunD \rangle \langle FunD \rangle ; \langle FunDF \rangle$	nicht-leere Funktionsdeklarationsfolge
$\langle FunD \rangle$	$\rightarrow \langle Na \rangle \langle Na \rangle (\langle PaDF \rangle) \{ \langle VarDF \rangle ; \langle rumpf \rangle \} $ $\langle Na \rangle \langle Na \rangle (\langle PaDF \rangle) \{ \langle rumpf \rangle \}$	Funktionsdeklaration mit lokalen Variablen Funktionsdeklaration ohne lokale Variablen
$\langle PaDF \rangle$	$\rightarrow \varepsilon \langle ParDF \rangle$	Parameterdeklarationsfolge
$\langle ParDF \rangle$	$\rightarrow \langle VarD \rangle \langle VarD \rangle , \langle PaDF \rangle$	nicht-leere Parameterdeklarationsfolge
$\langle rumpf \rangle$	$\rightarrow \langle rAn \rangle \langle AnF \rangle ; \langle rAn \rangle$	Funktionsrumpf

Tabelle 1: Grammatik von C0

$$e-consis(c, d) \equiv \forall xs \in SV(c) : (vtyp(c, xs) \in ET) \rightarrow d.m_4(ba(c, xs)) = zxt(c.m(xs))$$

$$p-consis(c, d) \equiv \forall xs \in SV(c) : vtyp(c, xs) = t^* \rightarrow d.m_4(ba(c, xs)) = ba(c, c.m(xs))$$

$$pc-consis(c, d) \equiv d.pc = start(code(\mathbf{hd}(c.pr)))$$

$$ra-consis(c, d) \equiv \forall i \in [1 : c.rd] : ra(c, d, i) = start(code(c.pr[ret(i, c) + 1]))$$

$$ret(0, c) = \max\{j \mid l(c.pr(j)) = \langle rAn \rangle\}, \quad ret(i, c) = \max\{j < ret(i-1, c) \mid l(c.pr(j)) = \langle rAn \rangle\}$$

$$hsp-consis(c, d) \equiv d.gpr(hp) = ba(c.nh-1, c) +_{32} size(c.ht(c.nh-1))_{32} \wedge \langle d.gpr(hp) \rangle < \langle hmax \rangle$$

$$\wedge d.gpr(sp) = ba(top(c), c) +_{32} size(\$c.st(c.rd))_{32} +_{32} 4_{32} \wedge \langle d.gpr(sp) \rangle < \langle hbase \rangle$$

$$consis(c, d) \equiv e-consis(c, d) \wedge p-consis(c, d) \wedge pc-consis(c, d) \wedge ra-consis(c, d) \wedge hsp-consis(c, d) \wedge d.gpr(28_5) = sbase$$

$$Ra(t) = \begin{cases} \mathbb{B}^{32} & t \in \{uint, int\} \\ \mathbb{B}^8 & t = char \\ \mathbb{B} & t = bool \\ \{f : [0 : n-1] \rightarrow Ra(t')\} & tt(t) = t'[n] \\ \{f \mid f : \{n_1, \dots, n_s\} \rightarrow Ra(t_1) \cup \dots \cup Ra(t_s), \forall i : f(n_i) \in Ra(t_i)\} & tt(t) = \{t_1 \ n_1; \dots; t_s \ n_s\} \\ VS^* \cup \{null\} & tt(t) = t^* \end{cases}$$

I-Type Data Transfer		R-Type Shift Operation	
<i>lb rt rs imm</i>	$rt = \text{sxt}(m_1(ea(c)))$	<i>sll rd rt sa</i>	$rd = \text{sll}(rt,sa)$
<i>lh rt rs imm</i>	$rt = \text{sxt}(m_2(ea(c)))$	<i>srl rd rt sa</i>	$rd = \text{srl}(rt,sa)$
<i>lw rt rs imm</i>	$rt = m_4(ea(c))$	<i>sra rd rt sa</i>	$rd = \text{sra}(rt,sa)$
<i>lbu rt rs imm</i>	$rt = 0^{24}m_1(ea(c))$	<i>sllv rd rt rs</i>	$rd = \text{sll}(rt,rs)$
<i>lhu rt rs imm</i>	$rt = 0^{16}m_2(ea(c))$	<i>srlv rd rt rs</i>	$rd = \text{srl}(rt,rs)$
<i>sb rt rs imm</i>	$m_1(ea(c)) = rt[7:0]$	<i>srav rd rt rs</i>	$rd = \text{sra}(rt,rs)$
<i>sh rt rs imm</i>	$m_2(ea(c)) = rt[15:0]$	R-Type Arithmetic, Logical Operation	
<i>sw rt rs imm</i>	$m_4(ea(c)) = rt$	<i>add rd rs rt</i>	$rd = rs + rt$
I-Type Arithmetic, Logical Operation, Test-and-Set		<i>addu rd rs rt</i>	$rd = rs + rt$
<i>addi rt rs imm</i>	$rt = rs + \text{sxt}(imm)$	<i>sub rd rs rt</i>	$rd = rs - rt$
<i>addiu rt rs imm</i>	$rt = rs + \text{sxt}(imm)$	<i>subu rd rs rt</i>	$rd = rs - rt$
<i>slti rt rs imm</i>	$rt = ([rs] < [\text{sxt}(imm)] ? 1 : 0)$	<i>and rd rs rt</i>	$rd = rs \wedge rt$
<i>sltui rt rs imm</i>	$rt = ((rs) < (\text{sxt}(imm)) ? 1 : 0)$	<i>or rd rs rt</i>	$rd = rs \vee rt$
<i>andi rt rs imm</i>	$rt = rs \wedge \text{zxt}(imm)$	<i>xor rd rs rt</i>	$rd = rs \oplus rt$
<i>ori rt rs imm</i>	$rt = rs \vee \text{zxt}(imm)$	<i>nor rd rs rt</i>	$rd = rs \bar{\vee} rt$
<i>xori rt rs imm</i>	$rt = rs \oplus \text{zxt}(imm)$	R-Type Test Set Operation	
<i>lui rt imm</i>	$rt = \text{imm}0^{16}$	<i>slt rd rs rt</i>	$rd = ([rs] < [rt] ? 1 : 0)$
I-Type Branch		<i>sltu rd rs rt</i>	$rd = ((rs) < (rt) ? 1 : 0)$
<i>bltz rs imm</i>	$pc = pc + (rs < 0 ? \text{imm}00 : 4)$	R-Type Jumps, System Call	
<i>bgez rs imm</i>	$pc = pc + (rs \geq 0 ? \text{imm}00 : 4)$	<i>jr rs</i>	$pc = rs$
<i>beq rs rt imm</i>	$pc = pc + (rs = rt ? \text{imm}00 : 4)$	<i>jalr rd rs</i>	$rd = pc + 4 \quad pc = rs$
<i>bne rs rt imm</i>	$pc = pc + (rs \neq rt ? \text{imm}00 : 4)$	<i>sysc</i>	System Call
<i>blez rs imm</i>	$pc = pc + (rs \leq 0 ? \text{imm}00 : 4)$	Coprocessor Instructions	
<i>bgtz rs imm</i>	$pc = pc + (rs > 0 ? \text{imm}00 : 4)$	<i>eret</i>	Exception Return
J-Type Jumps		<i>movg2s rd rt</i>	$\text{spr}[rd] := \text{gpr}[rt]$
<i>j iindex</i>	$pc = \text{bin}_{32}(pc+4)[31:28]iindex00$	<i>movs2g rd rt</i>	$\text{gpr}[rt] := \text{spr}[rd]$
<i>jal iindex</i>	$R31 = pc + 4$ $pc = \text{bin}_{32}(pc+4)[31:28]iindex00$		

Tabelle 2: Assembler-Syntax MIPS, es gilt: $ea(c) = rs(c) +_{32} \text{sxtimm}(c)$

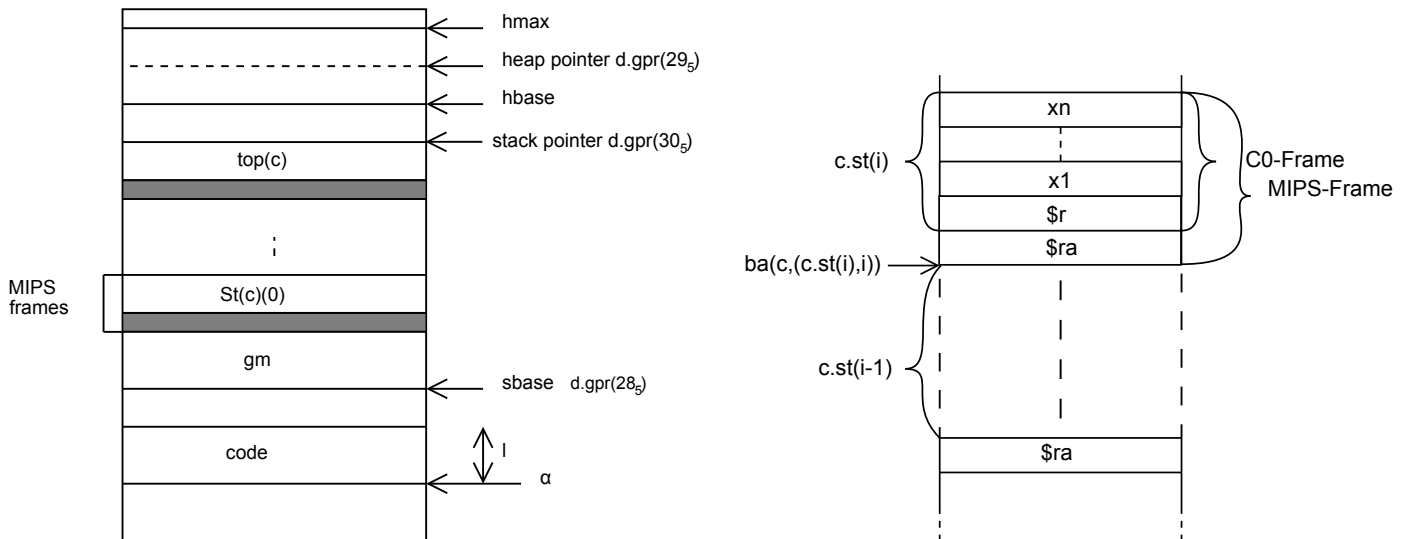


Abbildung 1: Memory-Layout + Stack-Layout des C0-Compilers

<i>i</i>				maskierbar	
0	reset	eev	abort	0	
1-15	I/O	eev	repeat	1	Input/Output (Devices)
16	ill	iev	abort	0	illegal instruction
17	mal	iev	abort	0	misaligned
18	pff	iev	repeat	0	page fault fetch
19	pfls	iev	repeat	0	page fault load/store
20	sysc	iev	continue	0	system call
21	ovf	iev	continue	1	overflow

Tabelle 3: Interrupts der MIPS

<i>i</i>	spr(i)	
0	sr	status register
1	esr	exception sr
2	eca	exception cause register
3	epc	exception pc
4	edata	exception data
5	pto	page table origin
6	ptl	page table length
7	mode	mode register

Tabelle 4: Special-Purpose-Register der MIPS