$\langle Zi \rangle$	$\longrightarrow$	$0 \mid \ldots \mid 9$	Ziffer
$\langle ZiF \rangle$	$\longrightarrow$	$\langle Zi \rangle \mid \langle ZiF \rangle \langle Zi \rangle$	Ziffernfolge
$\langle Bu \rangle$	$\longrightarrow$	$a \mid \ldots \mid z \mid A \mid \ldots \mid Z$	Buchstabe
$\langle BuZi \rangle$	$\longrightarrow$	$\langle Bu \rangle \mid \langle Zi \rangle$	alphanumerisches Zeichen
$\langle BuZiF \rangle$	$\longrightarrow$	$\langle BuZi \rangle \mid \langle BuZiF \rangle \langle BuZi \rangle$	alphanumerische Zeichenfolge
$\langle Na \rangle$	$\longrightarrow$	$\langle Bu \rangle \mid \langle Bu \rangle \langle BuZiF \rangle$	Name
$\langle C \rangle$	$\longrightarrow$	$\langle ZiF \rangle$	*int-/uint*-Konstante
$\langle CC \rangle$	$\longrightarrow$	'$\langle BuZi \rangle$'	*char*-Konstante
$\langle BC \rangle$	$\longrightarrow$	$0 \mid 1$	*bool*-Konstante
$\langle id \rangle$	$\longrightarrow$	$\langle Na \rangle \mid \langle id \rangle.\langle Na \rangle \mid \langle id \rangle[\langle A \rangle] \mid \langle id \rangle* \mid \langle id \rangle\&$	Identifier ((Sub-)Variablenbezeichner)
$\langle F \rangle$	$\longrightarrow$	$\langle id \rangle \mid -_1\langle F \rangle \mid (\langle A \rangle) \mid \langle C \rangle$	Faktor
$\langle T \rangle$	$\longrightarrow$	$\langle F \rangle \mid \langle T \rangle \cdot \langle F \rangle \mid \langle T \rangle / \langle F \rangle$	Term
$\langle A \rangle$	$\longrightarrow$	$\langle T \rangle \mid \langle A \rangle + \langle T \rangle \mid \langle A \rangle -_2 \langle T \rangle$	(algebraischer) Ausdruck
$\langle Atom \rangle$	$\longrightarrow$	$\langle A \rangle > \langle A \rangle \mid \langle A \rangle \geq \langle A \rangle \mid \langle A \rangle < \langle A \rangle \mid \langle A \rangle \leq \langle A \rangle \mid$ $\langle A \rangle == \langle A \rangle \mid \langle A \rangle \neq \langle A \rangle \mid \langle id \rangle \neq null \mid \langle BC \rangle$	"Boolesche Variable"
$\langle BF \rangle$	$\longrightarrow$	$\langle id \rangle \mid \langle Atom \rangle \mid\sim \langle BF \rangle \mid (\langle BA \rangle)$	Boolescher Faktor
$\langle BT \rangle$	$\longrightarrow$	$\langle BF \rangle \mid \langle BT \rangle \wedge \langle BF \rangle$	Boolescher Term
$\langle BA \rangle$	$\longrightarrow$	$\langle BT \rangle \mid \langle BA \rangle \vee \langle BT \rangle$	Boolescher Ausdruck
$\langle An \rangle$	$\longrightarrow$	$\langle id \rangle = \langle A \rangle \mid \langle id \rangle = \langle BA \rangle \mid \langle id \rangle = \langle CC \rangle \mid$	Zuweisung
		$if\ \langle BA \rangle\ then\ \{\langle AnF \rangle\}\ else\ \{\langle AnF \rangle\} \mid$	bedingte Anweisung
		$if\ \langle BA \rangle\ then\ \{\langle AnF \rangle\} \mid$	
		$while\ \langle BA \rangle\ do\ \{\langle AnF \rangle\} \mid$	Schleife
		$\langle id \rangle = \langle Na \rangle(\langle PaF \rangle) \mid$	Funktionsaufruf mit Parametern
		$\langle id \rangle = new\ \langle Na \rangle*$	Speicher allozieren
$\langle rAn \rangle$	$\longrightarrow$	$return\ \langle A \rangle \mid return\ \langle BA \rangle$	Return-Anweisung
$\langle PaF \rangle$	$\longrightarrow$	$\varepsilon \mid \langle ParF \rangle$	Parameterfolge
$\langle ParF \rangle$	$\longrightarrow$	$\langle A \rangle \mid \langle ParF \rangle, \langle A \rangle$	nicht-leere Parameterfolge
$\langle AnF \rangle$	$\longrightarrow$	$\langle An \rangle \mid \langle AnF \rangle; \langle An \rangle$	Anweisungsfolge
$\langle program \rangle$	$\longrightarrow$	$\langle TypDF \rangle; \langle VarDF \rangle; \langle FunDF \rangle \mid$	$C0$-Programm
		$\langle VarDF \rangle; \langle FunDF \rangle \mid$	ohne Typdeklarationen
		$\langle TypDF \rangle; \langle FunDF \rangle \mid$	ohne Variablendeklarationen
		$\langle FunDF \rangle$	ohne Typ- und Variablendeklarationen
$\langle TypDF \rangle$	$\longrightarrow$	$\langle TypD \rangle \mid \langle TypD \rangle; \langle TypDF \rangle$	Typdeklarationsfolge, nicht-leer
$\langle TypD \rangle$	$\longrightarrow$	$typedef\ \langle Typ \rangle\ \langle Na \rangle$	Typdeklaration
$\langle Typ \rangle$	$\longrightarrow$	$\langle Na \rangle[\langle ZiF \rangle] \mid \langle Na \rangle* \mid struct\{\langle VarDF \rangle\}$	Array-Typ, Pointer, struct-Typ
$\langle VarDF \rangle$	$\longrightarrow$	$\langle VarD \rangle \mid \langle VarD \rangle; \langle VarDF \rangle$	Variablendeklarationsfolge, nicht-leer
$\langle VarD \rangle$	$\longrightarrow$	$\langle Na \rangle\ \langle Na \rangle$	Variablendeklaration
$\langle FunDF \rangle$	$\longrightarrow$	$\langle FunD \rangle \mid \langle FunDF \rangle; \langle FunD \rangle$	nicht-leere Funktionsdeklarationsfolge
$\langle FunD \rangle$	$\longrightarrow$	$\langle Na \rangle\ \langle Na \rangle(\langle PaDF \rangle)\{\langle VarDF \rangle; \langle rumpf \rangle\} \mid$	Funktionsdeklaration mit lokalen Variablen
		$\langle Na \rangle\ \langle Na \rangle(\langle PaDF \rangle)\{\langle rumpf \rangle\}$	Funktionsdeklaration ohne lokale Variablen
$\langle PaDF \rangle$	$\longrightarrow$	$\varepsilon \mid \langle ParDF \rangle$	Parameterdeklarationfolge
$\langle ParDF \rangle$	$\longrightarrow$	$\langle VarD \rangle \mid \langle VarD \rangle, \langle PaDF \rangle$	nicht-leere Parameterdeklarationfolge
$\langle rumpf \rangle$	$\longrightarrow$	$\langle rAn \rangle \mid \langle AnF \rangle; \langle rAn \rangle$	Funktionsrumpf

Tabelle 1: Grammatik von $C0$

I-Type Data Transfer	
lb $rt\ rs\ imm$	rt = sxt($m_1(ea(c))$)
lh $rt\ rs\ imm$	rt = sxt($m_2(ea(c))$)
lw $rt\ rs\ imm$	rt = $m_4(ea(c))$
lbu $rt\ rs\ imm$	rt = $0^{24}m_1(ea(c))$
lhu $rt\ rs\ imm$	rt = $0^{16}m_2(ea(c))$
sb $rt\ rs\ imm$	$m_1(ea(c))$ = rt[7:0]
sh $rt\ rs\ imm$	$m_2(ea(c))$ = rt[15:0]
sw $rt\ rs\ imm$	$m_4(ea(c))$ = rt
I-Type Arithmetic, Logical Operation, Test-and-Set	
addi $rt\ rs\ imm$	rt = rs + sxt(imm)
addiu $rt\ rs\ imm$	rt = rs + sxt(imm)
slti $rt\ rs\ imm$	rt = (rs < sxt(imm) ? 1 : 0)
sltui $rt\ rs\ imm$	rt = (rs < sxt(imm) ? 1 : 0)
andi $rt\ rs\ imm$	rt = rs ∧ zxt(imm)
ori $rt\ rs\ imm$	rt = rs ∨ zxt(imm)
xori $rt\ rs\ imm$	rt = rs ⊕ zxt(imm)
lui $rt\ imm$	rt = imm0^{16}
I-Type Branch	
bltz $rs\ imm$	pc = pc + (rs < 0 ? imm00 : 4)
bgez $rs\ imm$	pc = pc + (rs ≥ 0 ? imm00 : 4)
beq $rs\ rt\ imm$	pc = pc + (rs = rt ? imm00 : 4)
bne $rs\ rt\ imm$	pc = pc + (rs ≠ rt ? imm00 : 4)
blez $rs\ imm$	pc = pc + (rs ≤ 0 ? imm00 : 4)
bgtz $rs\ imm$	pc = pc + (rs > 0 ? imm00 : 4)
J-Type Jumps	
j $iindex$	pc = bin_{32}(pc+4)[31:28]iindex00
jal $iindex$	R31 = pc + 4
	pc = bin_{32}(pc+4)[31:28]iindex00

R-Type Shift Operation	
sll $rd\ rt\ sa$	rd = sll(rt,sa)
srl $rd\ rt\ sa$	rd = srl(rt,sa)
sra $rd\ rt\ sa$	rd = sra(rt,sa)
sllv $rd\ rt\ rs$	rd = sll(rt,rs)
srlv $rd\ rt\ rs$	rd = srl(rt,rs)
srav $rd\ rt\ rs$	rd = sra(rt,rs)
R-Type Arithmetic, Logical Operation	
add $rd\ rs\ rt$	rd = rs + rt
addu $rd\ rs\ rt$	rd = rs + rt
sub $rd\ rs\ rt$	rd = rs − rt
subu $rd\ rs\ rt$	rd = rs − rt
and $rd\ rs\ rt$	rd = rs ∧ rt
or $rd\ rs\ rt$	rd = rs ∨ rt
xor $rd\ rs\ rt$	rd = rs ⊕ rt
nor $rd\ rs\ rt$	rd = rs $\overline{\vee}$ rt
R-Type Test Set Operation	
slt $rd\ rs\ rt$	rd = (rs < rt ? 1 : 0)
sltu $rd\ rs\ rt$	rd = (rs < rt ? 1 : 0)
R-Type Jumps, System Call	
jr rs	pc = rs
jalr $rd\ rs$	rd = pc + 4 pc = rs
sysc	System Call
Coprocessor Instructions	
eret	Exception Return
movg2s $rd\ rt$	spr[rd] := gpr[rt]
movs2g $rd\ rt$	gpr[rt] := spr[rd]

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

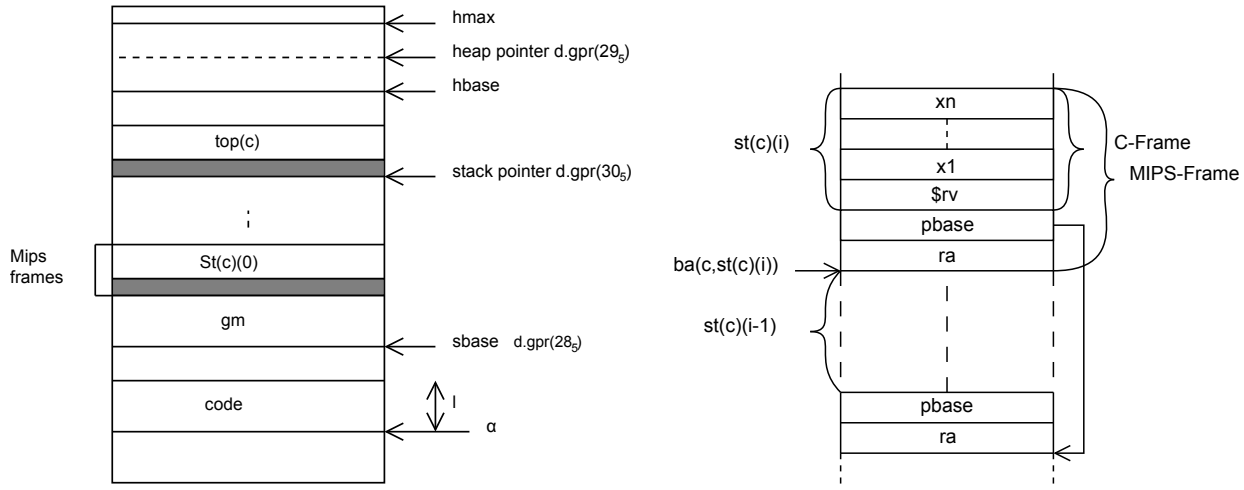

Abbildung 1: Memory-Layout + Stack-Layout des C0-Compilers

$\mathit{e\text{-}consis}(c,d) \equiv \forall xs \in SV(c) : (vtyp(c,xs) \in ET) \rightarrow d.m_4(ba(c,xs)) = zxt(c.m(xs))$
$\mathit{p\text{-}consis}(c,d) \equiv \forall xs \in SV(c) : vtyp(c,xs) = t* \rightarrow d.m_4(ba(c,xs)) = ba(c,c.m(xs))$
$\mathit{c\text{-}consis}(c,d) \equiv d.pc = start(\mathbf{hd}(c.lpr))$
$\mathit{r\text{-}consis}(c,d) \equiv \forall i > 0 : pbase(c,d,i) = ba(c,st(c)(i-1)) \wedge \forall i > 0 : ra(c,d,i) = end(c.scalls(i)) +_{32} 4_{32}$
$\mathit{pi\text{-}consis}(d) \equiv d.m_l(\alpha) = d^0.m_l(\alpha), \qquad \alpha$ Startadresse des Codebereichs, l Länge des Codebereichs.
$\mathit{hsp\text{-}consis}(c,d) \equiv d.gpr(29_5) = ba(c,c.nh-1) + size(vtyp(c,c.nh-1))$
$\wedge d.gpr(30_5) = ba(c,top(c))$
$limits(d) \equiv \alpha + l \leq sbase$
$\wedge \langle d.gpr(30_5) \rangle + size(\$top(c)_1) + 8 < hbase$
$\wedge \langle d.gpr(29_5) \rangle < hmax$
$consis(c,d) \equiv \mathit{e\text{-}consis}(c,d) \wedge \mathit{p\text{-}consis}(c,d) \wedge \mathit{c\text{-}consis}(c,d) \wedge \mathit{r\text{-}consis}(c,d) \wedge \mathit{pi\text{-}consis}(d)$
$\wedge \mathit{hsp\text{-}consis}(c,d) \wedge limits(d) \wedge d.gpr(28_5) = sbase$

Tabelle 3: Compiler-Konsistenzrelationen

$$\mathbb{N} = \mathbb{N}_0 = \{0,1,2,\ldots\}$$
$$\mathbb{N}_+ = \{1,2,3,\ldots\}$$

Tabelle 4: Natürliche Zahlen

i				maskierbar	
0	reset	eev	abort	0	
1	ill	iev	abort	0	illegal instruction
2	mal	iev	abort	0	misaligned
3	pff	iev	repeat	0	page fault fetch
4	pfls	iev	repeat	0	page fault load/store
5	sysc	iev	continue	0	system call
6	ovf	iev	continue	1	overflow
7	I/O	eev	continue	1	

Tabelle 5: Interrupts der MIPS

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 6: Special-Purpose-Register der MIPS