

Conversiones

Diseño de Compiladores I



CONVERSIONES EN REPRESENTACIONES INTERMEDIAS

Tipos

$$w := x + y * z$$

Con:

w de tipo double

x de tipo integer

y de tipo float

z de tipo long

Árbol Sintáctico con tipos y conversiones

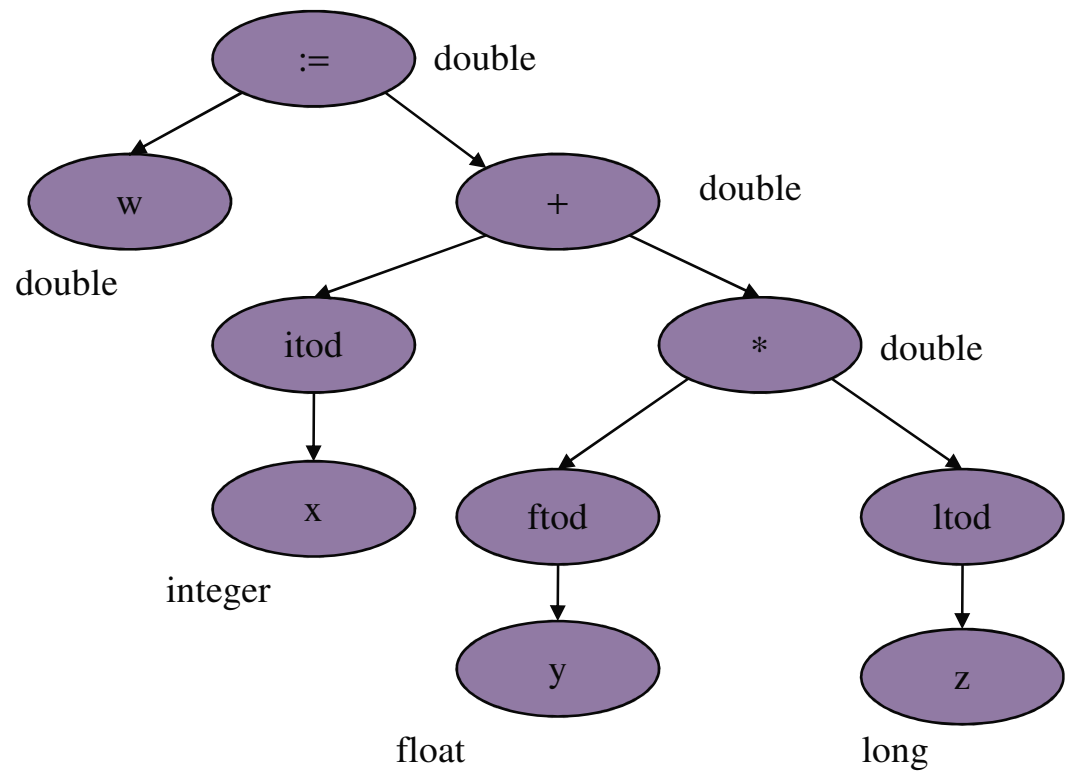
$W := X + Y * Z$

w de tipo double

x de tipo integer

y de tipo float

z de tipo long



Tercetos

con tipos y conversiones

$w := x + y * z$

w de tipo double

x de tipo integer

y de tipo float

z de tipo long

20. (ftod , y , -) *double*

21. (ltod , z , -) *double*

22. (* , [20] , [21]) *double*

23. (itod , x , -) *double*

24. (+ , [22] , [23]) *double*

25. (:= , w , [24])

Polaca Inversa

(desde Lista de Reglas)

$w := x + y * z$

w de tipo double

x de tipo integer

y de tipo float

z de tipo long

Polaca Inversa sin tipos ni conversiones



CONVERSIONES EN ASSEMBLER

Sin signo – Sobre registros

Conversión	Dato	Instrucciones	Resultado
8 a 16 bits	BL	MOV BH, 0	BX
16 a 32 bits	BX	MOV ECX, 0 MOV CX, BX MOV EBX, ECX	EBX
32 a 64 bits Para multiplicación y división	EAX	MOV EDX, 0	EDX : EAX

Sin signo – Sobre memoria

Datos	Tamaños
D1 DB ?	8 bits
D2 DW ?	16 bits
D4 DD ?	32 bits
D8 DQ ?	64 bits
D10 DT ?	80 bits

Sin signo – Sobre memoria

Conversión	Dato	Instrucciones	Resultado
8 a 16 bits	D1	MOV AL, D1 MOV D2, AL MOV AL, 0 MOV D2 + 1, AL	 <p>D2</p>
16 a 32 bits	D2	MOV AX, D2 MOV D4, AX MOV AX, 0 MOV D4 + 2, AX	 <p>D4</p>
32 a 64 bits	D4	MOV EAX, D4 MOV D4, EAX MOV EAX, 0 MOV D8 + 4, EAX	 <p>D8</p>

Sin signo – Sobre registros

(con control de overflow)

Conversión	Dato	Instrucciones	Resultado
16 a 8 bits	AX	CMP AH, 0 JNZ ERROR MOV BL, AL ERROR: ...	BL

Sin signo – Sobre registros

(con control de overflow)

Conversión	Dato	Instrucciones	Resultado
32 a 16 bits	EAX	MOV EBX, 0 MOV BX, AX CMP EAX, EBX JNE ERROR MOV CX, AX ERROR: ...	CX

Sin signo – Sobre memoria

(con control de overflow)

Conversión	Dato	Instrucciones	Resultado
16 a 8 bits	D2	MOV AL, D2 + 1 CMP AL, 0 JNE ERROR MOV AL, D2 MOV D1, AL ERROR: ...	D1

Sin signo – Sobre memoria

(con control de overflow)

Conversión	Dato	Instrucciones	Resultado
32 a 16 bits	D4	MOV AX, D4 + 2 CMP AX, 0 JNE ERROR MOV AX, D4 MOV D2, AX ERROR: ...	D2

Con signo (datos enteros)

Conversión	Dato	Instrucciones	Resultado
8 a 16 bits	CL	MOV AL, CL CBW	AX
16 a 32 bits	BX	MOV AX, BX CWD	DX:AX
16 a 32 bits	DX	MOV AX, DX CWDE	EAX
32 a 64 bits	EBX	MOV EAX, EBX CDQ	EDX:EAX

Con signo (datos enteros)

Conversión	Dato	Instrucciones	Resultado
32 a 64 bits Para multiplicaciones y divisiones	ECX	MOV EAX, ECX CMP EAX, 0 JL NEGATIVO MOV EDX, 0 JMP SIGO NEGATIVO: MOV EDX, -1 SIGO: ...	EDX : EAX

Con signo

(datos enteros con control de overflow)

Conversión	Dato	Instrucciones		Resultado
16 a 8 bits	AX	CMP AX, 0 JL NEGATIVO CMP AH, 0 JNE ERROR CMP AL, 0 JL ERROR JMP SIGO	NEGATIVO: CMP AH, -1 JNE ERROR CMP AL, 0 JG ERROR ERROR: ... SIGO: ...	AL

Datos flotantes

(Registro a Memoria)

Conversión	Dato	Instrucciones	Resultado
80 a 32 bits 64 a 32 bits	ST(0)	FST D4 o FSTP D4	D4 (32 bits)
80 a 64 bits 32 a 64 bits	ST(0)	FST D8 o FSTP D8	D8 (64 bits)
64 a 80 bits 32 a 80 bits	ST(0)	FST D10 o FSTP D10	D10 (80 bits)

Datos flotantes

(Memoria a Memoria)

Conversión	Dato	Instrucciones	Resultado
32 a 64 bits	D4	FLD D4 FST D8	D8 (64 bits)
64 a 32 bits	D8	FLD D8 FST D4	D4 (32 bits)
32 a 80 bits	D4	FLD D4 FSTP D10	D10 (80 bits)
...			

Conversiones entre datos enteros y flotantes

Conversión	Dato	Instrucciones	Resultado
Entero a float	D2 (entero de 16 bits)	FILD D2	ST(0) (flotante)
Entero a float	D4 (entero de 32 bits)	FILD D4	ST(0) (flotante)
Entero a float	D8 (entero de 64 bits)	FILD D8	ST(0) (flotante)

Conversiones entre datos enteros y flotantes

Conversión	Dato	Instrucciones	Resultado
Float a entero	ST(0) (flotante)	FIST D2 o FISTP D2	D2 (entero de 16 bits)
Float a entero	ST(0) (flotante)	FIST D4 o FISTP D4	D4 (entero de 32 bits)
Float a entero	ST(0) (flotante)	FIST D8 o FISTP D8	D8 (entero de 64 bits)

Conversiones entre datos con y sin signo

- Origen: con signo – Destino: sin signo
 - Origen positivo, se puede
 - Origen negativo, es error
- Origen: sin signo – Destino: con signo
 - Se debe testear overflow
 - Primer bit = 0, se puede
 - Primer bit = 1, es overflow