

L'ASSEMBLEUR – PARTIE 2

Xavier Merrheim

Traduire un if ... else ...

```
int a,b,c
```

```
main()
```

```
{
```

```
a=10;b=20;
```

```
if(a>b)c=a else c=b+3;
```

```
c=c-b;
```

```
}
```

Traduction en assembleur ARM

L1:	ldr r0,L1	add r1,r1,#3
.word a	ldr r1,[r0]	ldr r0,L1+8
.word b	ldr r0,L1+4	str r1,[r0]
.word c	ldr r2,[r0]	fin_if: ldr r0,L1+8
.comm a,4,4	cmp r1,r2	ldr r1,[r0]
.comm b,4,4	ble else	ldr r2,L1+4
.comm c,4,4	ldr r0,L1	ldr r3,[r2]
mov r0,#10	ldr r1,[r0]	sub r1,r1,r3
ldr r1,L1	ldr r0,L1+8	str r1,[r0]
str r1,[r0]	str r1,[r0]	
move r0,#20	bra fin_if	
ldr r1,L1+4	else : ldr r0,L1+4	
str r1,[r0]	ldr r1,[r0]	

Exercice : traduire en assembleur ARM

```
int a,b,c,d;
```

```
main()
```

```
{a=5;b=9;
```

```
if(b<a+3)c=a+b; else {c=b-a;a=a+1;}
```

```
d=c+b+90;
```

```
}
```

Solution

L1:	mov r0,#9	add r1,r1,r2	add r1,r1,#1
.word a	ldr r1,L1+4	ldr r0,L1+8	str r1,[r0]
.word b	str r0,[r1]	str r1,[r0]	fin: ldr r0,L1+8
.word c	ldr r0,L1+4	bra fin	ldr r1,[r0]
.word d	ldr r1,[r0]	ldr r0,L1+4	ldr r0,L1+4
.comm a,4,4	ldr r0,L1	ldr r1,[r0]	ldr r2,[r0]
.comm b,4,4	ldr r2,[r0]	ldr r0,L1	add r1,r1,r2
.comm c,4,4	add r2,r2,#3	ldr r2,[r0]	add r1,r1,#90
.comm d,4,4	bge else	sub r1,r1,r2	ldr r0,L1+12
mov r0,#5	ldr r0,L1	ldr r0,L1+8	str r1,[r0]
ldr r1,L1	ldr r1,[r0]	str r1,[r0]	
str r0,[r1]	ldr r0,L1+4	ldr r0,L1	
	ldr r2,[r0]	ldr r1,[r0]	

Traduire un for

```
int i,s;  
main()  
{  
s=0;  
for(i=1;i<10;i++)s=s+i;  
}
```

Traduction

```
L1:
.word i
.word s
mov r0,#0
ldr r1,L1+4
str r0,[r1]
mov r0,#1
ldr r1,L1
str r0,[r1]
for: ldr r0,L1
ldr r1,[r0]
cmp r1,#10
```

```
bge fin
ldr r0,L1+4
ldr r1,[r0]
ldr r2,L1
ldr r3,[r2]
add r1,r1,r3
str r3,[r0]
fin:
```

Exercice : traduire en assembleur ARM

```
int a,b,s,i;  
main()  
{a=1;b=2;s=0;  
for(i=1;i<=12;i++)  
    {  
        a=a+b;  
        b=b+1;  
        s=s+a;  
    }  
}
```


Solution

L1:	mov r0,#0	str r1,[r0]	str r1,[r0]
.word a	ldr r1,L1+8	ldr r0,L1+4	bra for
.word b	str r0,[r1]	ldr r1,[r0]	
.word s	mov r0,#1	add r1,r1,#1	
.word i	ldr r1,L1+12	str r1,[r0]	
.comm a,4,4	str r0,[r1]	ldr r0,L1+8	
.comm b,4,4	for: ldr r0,L1+12	ldr r1,[r0]	
.comm s,4,4	ldr r1,[r0]	ldr r2,L1	
.comm i,4,4	cmp r1,#12	ldr r3,[r2]	
mov r0,#1	bgt fin	add r1,r1,r3	
ldr r1,L1	ldr r0,L1	str r1,[r0]	
str r0,[r1]	ldr r1,[r0]	ldr r0,L1+12	
mov r0,#2	ldr r2,L1+4	ldr r1,[r0]	
ldr r1,L1+4	ldr r3,[r2]	add r1,r1,#1	
str r0,[r1]	add r1,r1,r3		

Exercice : Traduire en assembleur ARM

```
int a,b;  
main()  
{  
a=1;b=2;  
while(a<100)  
    {  
    b=a+b;  
    a=a+b;  
    }  
    a=a+10;  
}
```

Solution

```
L1:
.word a
.word b
.comm a,4,4
.comm b,4,4
mov r0,#1
ldr r1,L1
str r0,[r1]
mov r0,#2
ldr r1,L1+4
str r0,[r1]
```

```
while: ldr r0,L1
      ldr r1,[r0]
      cmp r1,#100
      bge fin
      ldr r0,L1
      ldr r1,[r0]
      ldr r0,L1+4
      ldr r2,[r0]
      add r1,r1,r2
      str r1,[r0]
      ldr r0,L1
      ldr r1,[r0]
```

```
      ldr r2,L1+4
      ldr r2,[r0]
      add r1,r1,r2
      str r1,[r0]
      bra while
fin : ldr r0,L1
      ldr r1,[r0]
      add r1,r1,#10
      str r1,[r0]
```