

		ADDR	DATA
		\$FF9C	
		\$FFA0	
		\$FFA4	
		\$FFA8	
SP	\$FFF4	\$FFAC	
		\$FFB0	
		\$FFB4	
FP	\$FFF8	\$FFB8	
		\$FFBC	
		\$FFC0	
PC	\$1100	\$FFC4	
		\$FFC8	
		\$FFCC	
LR	\$400	\$FFD0	
		\$FFD4	
		\$FFD8	
		\$FFDC	
		\$FFE0	
		\$FFE4	
		\$FFE8	
		\$FFEC	
		\$FFF0	
		\$FFF4	
		\$FFF8	

Frame attivo

E' stato appena eseguito il salto alla funzione MAIN con una chiamata a BL.
Il valore del PC è stato salvato nel registro LR (R14) e il PC è stato aggiornato.
Sono dentro MAIN

\$1100	STMFD	SP!, {FP,LR}
	MOV	FP, SP
	SUB	SP, SP, #16
\$110C		
\$1140	LDR	R2, [FP, #-8]
	LDR	R1, [FP, #-12]
	SUB	R0, FP, #16
	STMFD	SP!, {R0-R2}
\$1150	BL	R
\$1154	ADD	SP, SP, #12
\$1158		
	MOV	SP, FP
	LDMFD	SP!, {FP, PC}

```
int main()
{
  int W, X, Y, Z;
  R( X, Y, &Z );
}
```

Allocazione dinamica della memoria in ARM

		ADDR	DATA
		\$FF9C	
		\$FFA0	
		\$FFA4	
		\$FFA8	
SP	\$FFEC	\$FFAC	
		\$FFB0	
		\$FFB4	
FP	\$FFEC	\$FFB8	
		\$FFBC	
		\$FFC0	
PC	\$1108	\$FFC4	
		\$FFC8	
		\$FFCC	
LR	\$400	\$FFD0	
		\$FFD4	
		\$FFD8	
		\$FFDC	
		\$FFE0	
		\$FFE4	
		\$FFE8	
		\$FFEC	\$FFF8
		\$FFF0	\$400
		\$FFF4	
		\$FFF8	

Salvo il valore di LR nello stack, assieme al valore di FP.
Poi aggiorno il valore di FP: ho creato un nuovo frame

\$1100	STMFD	SP!, {FP,LR}
	MOV	FP, SP
	SUB	SP, SP, #16
\$110C		
\$1140	LDR	R2, [FP, #-8]
	LDR	R1, [FP, #-12]
	SUB	R0, FP, #16
	STMFD	SP!, {R0-R2}
\$1150	BL	R
\$1154	ADD	SP, SP, #12
\$1158		
	MOV	SP, FP
	LDMFD	SP!, {FP, PC}

```
int main()
{
  int W, X, Y, Z;
  R( X, Y, &Z );
}
```

Allocazione dinamica della memoria in ARM

		ADDR	DATA	
		\$FF9C		
		\$FFA0		
		\$FFA4		
		\$FFA8		
		\$FFAC		
SP	\$FFDC	\$FFB0		
		\$FFB4		
		\$FFB8		
FP	\$FFEC	\$FFBC		
		\$FFC0		
		\$FFC4		
PC	\$110C	\$FFC8		
		\$FFCC		
		\$FFD0		
LR	\$400	\$FFD4		
		\$FFD8		
		\$FFDC	0	Z
		\$FFE0	0	Y
		\$FFE4	0	X
		\$FFE8	0	W
		\$FFEC	\$FFF8	
		\$FFF0	\$400	
		\$FFF4		
		\$FFF8		

Ho allocato lo spazio per le variabili locali.
Terminata la creazione del frame

\$1100	STMFD	SP!, {FP,LR}	int main()
	MOV	FP, SP	{
	SUB	SP, SP, #16	int W, X, Y, Z;
\$110C			R(X, Y, &Z);
			}
\$1140	LDR	R2, [FP, #-8]	
	LDR	R1, [FP, #-12]	
	SUB	R0, FP, #16	
	STMFD	SP!, {R0-R2}	
\$1150	BL	R	
\$1154	ADD	SP, SP, #12	
\$1158			
	MOV	SP, FP	
	LDMFD	SP!, {FP, PC}	

Allocazione dinamica della memoria in ARM

		ADDR	DATA	
		\$FF9C		
		\$FFA0		
		\$FFA4		
		\$FFA8		
		\$FFAC		
SP	\$FFDC	\$FFB0		
		\$FFB4		
		\$FFB8		
FP	\$FFEC	\$FFBC		
		\$FFC0		
		\$FFC4		
PC	\$1140	\$FFC8		
		\$FFCC		
		\$FFD0		
LR	\$400	\$FFD4		
		\$FFD8		
		\$FFDC	3	Z
		\$FFE0	27	Y
		\$FFE4	111	X
		\$FFE8	12	W
		\$FFEC	\$FFF8	
		\$FFF0	\$400	
		\$FFF4		
		\$FFF8		

Vengono eseguite le istruzioni contenute in main (non esplicitate)
Le variabili locali assumono dei valori calcolati da queste istruzioni

\$1100	STMFD	SP!, {FP,LR}	int main()
	MOV	FP, SP	{
	SUB	SP, SP, #16	int W, X, Y, Z;
\$110C			R(X, Y, &Z);
			}
\$1140	LDR	R2, [FP, #-8]	
	LDR	R1, [FP, #-12]	
	SUB	R0, FP, #16	
	STMFD	SP!, {R0-R2}	
\$1150	BL	R	
\$1154	ADD	SP, SP, #12	
\$1158			
	MOV	SP, FP	
	LDMFD	SP!, {FP, PC}	

Allocazione dinamica della memoria in ARM

		ADDR	DATA	
		\$FF9C		
		\$FFA0		
		\$FFA4		
		\$FFA8		
		\$FFAC		
SP	\$FFD0	\$FFB0		
		\$FFB4		
		\$FFB8		
FP	\$FFEC	\$FFBC		
		\$FFC0		
		\$FFC4		
PC	\$1150	\$FFC8		
		\$FFCC		
		\$FFD0	\$FFDC	&Z
LR	\$400	\$FFD4	27	Y
		\$FFD8	111	X
		\$FFDC	3	Z
		\$FFE0	27	Y
		\$FFE4	111	X
		\$FFE8	12	W
		\$FFEC	\$FFF8	
		\$FFF0	\$400	
		\$FFF4		
		\$FFF8		

Mi preparo alla chiamata alla subroutine R
Nel frame vengono preparati i parametri per la funzione R

\$1100	STMFD	SP!, {FP,LR}
	MOV	FP, SP
	SUB	SP, SP, #16
\$110C		
\$1140	LDR	R2, [FP, #-8]
	LDR	R1, [FP, #-12]
	SUB	R0, FP, #16
	STMFD	SP!, {R0-R2}
\$1150	BL	R
\$1154	ADD	SP, SP, #12
\$1158		
	MOV	SP, FP
	LDMFD	SP!, {FP, PC}

```
int main()
{
  int W, X, Y, Z;
  R( X, Y, &Z );
}
```

Allocazione dinamica della memoria in ARM

		ADDR	DATA	
		\$FF9C		
		\$FFA0		
		\$FFA4		
		\$FFA8		
		\$FFAC		
SP	\$FFD0	\$FFB0		
		\$FFB4		
		\$FFB8		
FP	\$FFEC	\$FFBC		
		\$FFC0		
		\$FFC4		
PC	\$1200	\$FFC8		
		\$FFCC		
		\$FFD0	\$FFDC	&Z
LR	\$1154	\$FFD4	27	Y
		\$FFD8	111	X
		\$FFDC	3	Z
		\$FFE0	27	Y
		\$FFE4	111	X
		\$FFE8	12	W
		\$FFEC	\$FFF8	
		\$FFF0	\$400	
		\$FFF4		
		\$FFF8		

E' stato appena eseguito il salto (chiamata a BL)

Il valore del PC è stato salvato nel registro LR e il suo valore aggiornato.

Il PC è stato aggiornato: sono dentro R

\$1200	STMFD	SPI, {FP,LR}
	MOV	FP, SP
	SUB	SP, SP, #12
\$120C		
	LDR	R0, [FP, #16]
	LDR	R1, [FP, #12]
	ADD	R0, R0, R1
	LDR	R1, [FP, #8]
	STR	R0, [R1]
\$1238		
	LDR	R0, [FP, #-4]
	CMP	R0, #0
\$1248	BNE	\$1268
\$124C		
	LDR	R2, [FP, #-4]
	LDR	R1, [FP, #-8]
	SUB	R0, FP, #12
	STMFD	SPI, {R0-R2}
\$125C	BL	R
\$1260	ADD	SP, SP, #12
\$1264		
\$1268	MOV	SP, FP
	LDMFD	SPI, {FP, PC}

```
void R( int I, int J, int *O )
```

```
{
int A, B, C;
*O = I+J;
if( A == 0 )
R( A, B, &C );
}
```

Allocazione dinamica della memoria in ARM

		ADDR	DATA	
		\$FF9C		
		\$FFA0		
		\$FFA4		
		\$FFA8		
		\$FFAC		
SP	\$FFC8	\$FFB0		
		\$FFB4		
		\$FFB8		
FP	\$FFC8	\$FFBC		
		\$FFC0		
		\$FFC4		
PC	\$1208	\$FFC8	\$FFEC	FP'
		\$FFCC	\$1154	LR(PC)
		\$FFD0	\$FFDC	&Z
LR	\$1154	\$FFD4	27	Y
		\$FFD8	111	X
		\$FFDC	3	Z
		\$FFE0	27	Y
		\$FFE4	111	X
		\$FFE8	12	W
		\$FFEC	\$FFF8	
		\$FFF0	\$400	
		\$FFF4		
		\$FFF8		

Salvo il valore di LR nello stack, assieme al valore di FP.
Poi aggiorno il valore di FP: ho creato un nuovo frame.

\$1200	STMFD	SPI, {FP,LR}
	MOV	FP, SP
	SUB	SP, SP, #12
\$120C		
	LDR	R0, [FP, #16]
	LDR	R1, [FP, #12]
	ADD	R0, R0, R1
	LDR	R1, [FP, #8]
	STR	R0, [R1]
\$1238		
	LDR	R0, [FP, #-4]
	CMP	R0, #0
\$1248	BNE	\$1268
\$124C		
	LDR	R2, [FP, #-4]
	LDR	R1, [FP, #-8]
	SUB	R0, FP, #12
	STMFD	SPI, {R0-R2}
\$125C	BL	R
\$1260	ADD	SP, SP, #12
\$1264		
\$1268	MOV	SP, FP
	LDMFD	SPI, {FP, PC}

```
void R( int I, int J, int *O )  
{  
  int A, B, C;  
  *O = I+J;  
  if( A == 0 )  
    R( A, B, &C );  
}
```

Allocazione dinamica della memoria in ARM

	ADDR	DATA	
	\$FF9C		
	\$FFA0		
	\$FFA4		
	\$FFA8		
	\$FFAC		
SP	\$FFBC		
	\$FFB0		
	\$FFB4		
	\$FFB8		
FP	\$FFC8		
	\$FFBC	0	C
	\$FFC0	0	B
	\$FFC4	0	A
PC	\$120C	\$FFEC	FP'
	\$FFCC	\$1154	LR(PC)
	\$FFD0	\$FFDC	&Z
LR	\$1154	27	Y
	\$FFD4	111	X
	\$FFD8		
	\$FFDC	3	Z
	\$FFE0	27	Y
	\$FFE4	111	X
	\$FFE8	12	W
	\$FFEC	\$FFF8	
	\$FFF0	\$400	
	\$FFF4		
	\$FFF8		

Ho allocato lo spazio per le variabili locali
Terminata la creazione del frame

\$1200	STMFD	SPI, {FP,LR}	void R(int I, int J, int *O)
	MOV	FP, SP	{
	SUB	SP, SP, #12	int A, B, C;
\$120C			*O = I+J;
	LDR	R0, [FP, #16]	if(A == 0)
	LDR	R1, [FP, #12]	R(A, B, &C);
	ADD	R0, R0, R1	}
	LDR	R1, [FP, #8]	
	STR	R0, [R1]	
\$1238			
	LDR	R0, [FP, #-4]	
	CMP	R0, #0	
\$1248	BNE	\$1268	
\$124C	LDR	R2, [FP, #-4]	
	LDR	R1, [FP, #-8]	
	SUB	R0, FP, #12	
	STMFD	SPI, {R0-R2}	
\$125C	BL	R	
\$1260	ADD	SP, SP, #12	
\$1264			
\$1268	MOV	SP, FP	
	LDMFD	SPI, {FP, PC}	

Allocazione dinamica della memoria in ARM

		ADDR	DATA	
		\$FF9C		
		\$FFA0		
		\$FFA4		
		\$FFA8		
		\$FFAC		
SP	\$FFBC	\$FFB0		
		\$FFB4		
		\$FFB8		
FP	\$FFC8	\$FFBC	44	C
		\$FFC0	7	B
		\$FFC4	0	A
PC	\$1238	\$FFC8	\$FFEC	FP'
		\$FFCC	\$1154	LR(PC)
		\$FFD0	\$FFDC	&Z
LR	\$1154	\$FFD4	27	Y
		\$FFD8	111	X
		\$FFDC	138	Z
		\$FFE0	27	Y
		\$FFE4	111	X
		\$FFE8	12	W
		\$FFEC	\$FFF8	
		\$FFF0	\$400	
		\$FFF4		
		\$FFF8		

Eseguo le istruzioni in R (non esplicitate quelle che assegnano i valori ad A, B e C)
Viene assegnato al parametro di uscita O il valore I+J. Dato che O è stato passato per indirizzo, di fatto viene modificato direttamente il valore della variabile Z (variabile locale di MAIN, il cui indirizzo è in FP+8)

```

$1200  STMFD  SP!, {FP,LR}
        MOV   FP, SP
        SUB   SP, SP, #12

```

```

$120C
        LDR   R0, [FP, #16]
        LDR   R1, [FP, #12]
        ADD   R0, R0, R1
        LDR   R1, [FP, #8]
        STR   R0, [R1]

```

```

$1238
        LDR   R0, [FP, #-4]
        CMP   R0, #0
        BNE   $1248

```

```

$124C  LDR   R2, [FP, #-4]
        LDR   R1, [FP, #-8]
        SUB   R0, FP, #12
        STMFD SP!, {R0-R2}
        BL    R
$125C  BL    R
$1260  ADD   SP, SP, #12
$1264
$1268  MOV   SP, FP
        LDMFD SP!, {FP, PC}

```

```

void R( int I, int J, int *O )
{
    int A, B, C;
    *O = I+J;
    if( A == 0 )
        R( A, B, &C );
}

```

Allocazione dinamica della memoria in ARM

		ADDR	DATA	
		\$FF9C		
		\$FFA0		
		\$FFA4		
		\$FFA8		
		\$FFAC		
SP	\$FFBC	\$FFB0		
		\$FFB4		
		\$FFB8		
FP	\$FFC8	\$FFBC	44	C
		\$FFC0	7	B
		\$FFC4	0	A
PC	\$1248	\$FFC8	\$FFEC	FP'
		\$FFCC	\$1154	LR(PC)
		\$FFD0	\$FFDC	&Z
LR	\$1154	\$FFD4	27	Y
		\$FFD8	111	X
		\$FFDC	138	Z
		\$FFE0	27	Y
		\$FFE4	111	X
		\$FFE8	12	W
		\$FFEC	\$FFF8	
		\$FFF0	\$400	
		\$FFF4		
		\$FFF8		

Eseguo le istruzioni in R

La condizione A==0 è vera: viene eseguita la chiamata ricorsiva alla subroutine R

\$1200	STMFD	SPI, {FP,LR}
	MOV	FP, SP
	SUB	SP, SP, #12
\$120C		
	LDR	R0, [FP, #16]
	LDR	R1, [FP, #12]
	ADD	R0, R0, R1
	LDR	R1, [FP, #8]
	STR	R0, [R1]
\$1238		
	LDR	R0, [FP, #-4]
	CMP	R0, #0
\$1248	BNE	\$1268
\$124C	LDR	R2, [FP, #-4]
	LDR	R1, [FP, #-8]
	SUB	R0, FP, #12
	STMFD	SPI, {R0-R2}
\$125C	BL	R
\$1260	ADD	SP, SP, #12
\$1264		
\$1268	MOV	SP, FP
	LDMFD	SPI, {FP, PC}

```

void R( int I, int J, int *O )
{
  int A, B, C;
  *O = I+J;
  if( A == 0 )
    R( A, B, &C );
}

```

Allocazione dinamica della memoria in ARM

		ADDR	DATA	
		\$FF9C		
		\$FFA0		
		\$FFA4		
		\$FFA8		
		\$FFAC		
SP	\$FFB0	\$FFB0	\$FFBC	&C
		\$FFB4	7	B
		\$FFB8	0	A
FP	\$FFC8	\$FFBC	44	C
		\$FFC0	7	B
		\$FFC4	0	A
PC	\$125C	\$FFC8	\$FFEC	FP'
		\$FFCC	\$1154	LR(PC)
		\$FFD0	\$FFDC	&Z
LR	\$1154	\$FFD4	27	Y
		\$FFD8	111	X
		\$FFDC	138	Z
		\$FFE0	27	Y
		\$FFE4	111	X
		\$FFE8	12	W
		\$FFEC	\$FFF8	
		\$FFF0	\$400	
		\$FFF4		
		\$FFF8		

Mi preparo alla chiamata ricorsiva
Inserisco nello stack i valori dei parametri di ingresso della subroutine da chiamare

\$1200	STMFD	SPI, {FP,LR}
	MOV	FP, SP
	SUB	SP, SP, #12
\$120C		
	LDR	R0, [FP, #16]
	LDR	R1, [FP, #12]
	ADD	R0, R0, R1
	LDR	R1, [FP, #8]
	STR	R0, [R1]
\$1238		
	LDR	R0, [FP, #-4]
	CMP	R0, #0
\$1248	BNE	\$1268
\$124C	LDR	R2, [FP, #4]
	LDR	R1, [FP, #8]
	SUB	R0, FP, #12
	STMFD	SPI, {R0-R2}
\$125C	BL	R
\$1260	ADD	SP, SP, #12
\$1264		
\$1268	MOV	SP, FP
	LDMFD	SPI, {FP, PC}

```
void R( int I, int J, int *O )  
{  
  int A, B, C;  
  *O = I+J;  
  if( A == 0 )  
    R( A, B, &C );  
}
```

Allocazione dinamica della memoria in ARM

		ADDR	DATA	
		\$FF9C		
		\$FFA0		
		\$FFA4		
		\$FFA8		
		\$FFAC		
SP	\$FFB0	\$FFB0	\$FFBC	&C
		\$FFB4	7	B
		\$FFB8	0	A
FP	\$FFC8	\$FFBC	44	C
		\$FFC0	7	B
		\$FFC4	0	A
PC	\$1200	\$FFC8	\$FFEC	FP'
		\$FFCC	\$1154	LR(PC)
		\$FFD0	\$FFDC	&Z
LR	\$1260	\$FFD4	27	Y
		\$FFD8	111	X
		\$FFDC	138	Z
		\$FFE0	27	Y
		\$FFE4	111	X
		\$FFE8	12	W
		\$FFEC	\$FFF8	
		\$FFF0	\$400	
		\$FFF4		
		\$FFF8		

E' stato appena eseguito il salto (chiamata a BL)

Il valore del PC è stato salvato nel registro LR e il suo valore aggiornato.

Il PC è stato aggiornato: sono dentro R la seconda volta (ricorsione)

\$1200	STMFD	SPI, {FP,LR}
	MOV	FP, SP
	SUB	SP, SP, #12
\$120C		
	LDR	R0, [FP, #16]
	LDR	R1, [FP, #12]
	ADD	R0, R0, R1
	LDR	R1, [FP, #8]
	STR	R0, [R1]
\$1238		
	LDR	R0, [FP, #-4]
	CMP	R0, #0
\$1248	BNE	\$1268
\$124C		
	LDR	R2, [FP, #-4]
	LDR	R1, [FP, #-8]
	SUB	R0, FP, #12
	STMFD	SPI, {R0-R2}
\$125C	BL	R
\$1260	ADD	SP, SP, #12
\$1264		
\$1268	MOV	SP, FP
	LDMFD	SPI, {FP, PC}

```
void R( int I, int J, int *O )
```

```
{
int A, B, C;
*O = I+J;
if( A == 0 )
R( A, B, &C );
}
```

Allocazione dinamica della memoria in ARM

	ADDR	DATA	
	\$FF9C		
	\$FFA0		
	\$FFA4		
	\$FFA8	\$FFC8	FP"
	\$FFAC	\$1260	LR(PC)
SP	\$FFA8	\$FFB0	&C
		\$FFB4	7
		\$FFB8	0
FP	\$FFA8	\$FFBC	44
		\$FFC0	7
		\$FFC4	0
PC	\$1208	\$FFC8	\$FFEC
		\$FFCC	\$1154
		\$FFD0	\$FFDC
LR	\$1260	\$FFD4	27
		\$FFD8	111
		\$FFDC	138
		\$FFE0	27
		\$FFE4	111
		\$FFE8	12
		\$FFEC	\$FFF8
		\$FFF0	\$400
		\$FFF4	
		\$FFF8	

Salvo il valore di LR nello stack, assieme al valore di FP.
Poi aggiorno il valore di FP: ho creato un nuovo frame

\$1200	STMFD	SPI, {FP,LR}
	MOV	FP, SP
	SUB	SP, SP, #12
\$120C	LDR	R0, [FP, #16]
	LDR	R1, [FP, #12]
	ADD	R0, R0, R1
	LDR	R1, [FP, #8]
	STR	R0, [R1]
\$1238	LDR	R0, [FP, #-4]
	CMP	R0, #0
\$1248	BNE	\$1268
\$124C	LDR	R2, [FP, #-4]
	LDR	R1, [FP, #-8]
	SUB	R0, FP, #12
	STMFD	SPI, {R0-R2}
\$125C	BL	R
\$1260	ADD	SP, SP, #12
\$1264		
\$1268	MOV	SP, FP
	LDMFD	SPI, {FP, PC}

```
void R( int I, int J, int *O )
{
    int A, B, C;
    *O = I+J;
    if( A == 0 )
        R( A, B, &C );
}
```

Allocazione dinamica della memoria in ARM

		ADDR	DATA	
		\$FF9C	0	C
		\$FFA0	0	B
		\$FFA4	0	A
		\$FFA8	\$FFC8	FP"
		\$FFAC	\$1260	LR(PC)
SP	\$FF9C	\$FFB0	\$FFBC	&C
		\$FFB4	7	B
		\$FFB8	0	A
FP	\$FFA8	\$FFBC	44	C
		\$FFC0	7	B
		\$FFC4	0	A
PC	\$120C	\$FFC8	\$FFEC	FP'
		\$FFCC	\$1154	LR(PC)
		\$FFD0	\$FFDC	&Z
LR	\$1260	\$FFD4	27	Y
		\$FFD8	111	X
		\$FFDC	138	Z
		\$FFE0	27	Y
		\$FFE4	111	X
		\$FFE8	12	W
		\$FFEC	\$FFF8	
		\$FFF0	\$400	
		\$FFF4		
		\$FFF8		

Ho allocato lo spazio per le tre variabili locali

```

$1200  STMFD  SP!, {FP,LR}
        MOV   FP, SP
        SUB   SP, SP, #12
$120C

        LDR   R0, [FP, #16]
        LDR   R1, [FP, #12]
        ADD   R0, R0, R1
        LDR   R1, [FP, #8]
        STR   R0, [R1]

$1238

        LDR   R0, [FP, #-4]
        CMP   R0, #0
$1248    BNE   $1268

$124C    LDR   R2, [FP, #-4]
        LDR   R1, [FP, #-8]
        SUB   R0, FP, #12
        STMFD SP!, {R0-R2}
$125C    BL   R
$1260    ADD   SP, SP, #12
$1264
$1268    MOV   SP, FP
        LDMFD SP!, {FP, PC}
    
```

```

void R( int I, int J, int *O )
{
    int A, B, C;
    *O = I+J;
    if( A == 0 )
        R( A, B, &C );
}
    
```

Allocazione dinamica della memoria in ARM

		ADDR	DATA	
		\$FF9C	5	C
		\$FFA0	21	B
		\$FFA4	12	A
		\$FFA8	\$FFC8	FP"
		\$FFAC	\$1260	LR(PC)
SP	\$FF9C	\$FFB0	\$FFBC	&C
		\$FFB4	7	B
		\$FFB8	0	A
FP	\$FFA8	\$FFBC	7	C
		\$FFC0	7	B
		\$FFC4	0	A
PC	\$1238	\$FFC8	\$FFEC	FP'
		\$FFCC	\$1154	LR(PC)
		\$FFD0	\$FFDC	&Z
LR	\$1260	\$FFD4	27	Y
		\$FFD8	111	X
		\$FFDC	138	Z
		\$FFE0	27	Y
		\$FFE4	111	X
		\$FFE8	12	W
		\$FFEC	\$FFF8	
		\$FFF0	\$400	
		\$FFF4		
		\$FFF8		

Eseguo le istruzioni in R (non esplicitate quelle che assegnano i valori ad A, B e C)
Viene assegnato al parametro di uscita O il valore I+J. Dato che O è stato passato per indirizzo, di fatto viene modificato direttamente il valore della variabile C
(variabile locale della prima istanza di R, il cui indirizzo è in FP+8)

*O	\$1200	STMFD	SPI, {FP,LR}	void R(int I, int J, int *O)
J		MOV	FP, SP	{
I		SUB	SP, SP, #12	int A, B, C;
	\$120C			*O = I+J;
		LDR	R0, [FP, #16]	if(A == 0)
		LDR	R1, [FP, #12]	R(A, B, &C);
		ADD	R0, R0, R1	}
		LDR	R1, [FP, #8]	
		STR	R0, [R1]	
	\$1238			
		LDR	R0, [FP, #-4]	
		CMP	R0, #0	
	\$1248	BNE	\$1268	
	\$124C	LDR	R2, [FP, #-4]	
		LDR	R1, [FP, #-8]	
		SUB	R0, FP, #12	
		STMFD	SPI, {R0-R2}	
	\$125C	BL	R	
	\$1260	ADD	SP, SP, #12	
	\$1264			
	\$1268	MOV	SP, FP	
		LDMFD	SPI, {FP, PC}	

Allocazione dinamica della memoria in ARM

		ADDR	DATA	
		\$FF9C	5	C
		\$FFA0	21	B
		\$FFA4	12	A
		\$FFA8	\$FFC8	FP"
		\$FFAC	\$1260	LR(PC)
SP	\$FF9C	\$FFB0	\$FFBC	&C
		\$FFB4	7	B
		\$FFB8	0	A
FP	\$FFA8	\$FFBC	7	C
		\$FFC0	7	B
		\$FFC4	0	A
PC	\$1248	\$FFC8	\$FFEC	FP'
		\$FFCC	\$1154	LR(PC)
		\$FFD0	\$FFDC	&Z
LR	\$1260	\$FFD4	27	Y
		\$FFD8	111	X
		\$FFDC	138	Z
		\$FFE0	27	Y
		\$FFE4	111	X
		\$FFE8	12	W
		\$FFEC	\$FFF8	
		\$FFF0	\$400	
		\$FFF4		
		\$FFF8		

Viene valutata la condizione A==0

\$1200	STMFD	SPI, {FP,LR}
	MOV	FP, SP
	SUB	SP, SP, #12
\$120C		
	LDR	R0, [FP, #16]
	LDR	R1, [FP, #12]
	ADD	R0, R0, R1
	LDR	R1, [FP, #8]
	STR	R0, [R1]
\$1238		
	LDR	R0, [FP, #-4]
	CMP	R0, #0
\$1248	BNE	\$1268
\$124C	LDR	R2, [FP, #-4]
	LDR	R1, [FP, #-8]
	SUB	R0, FP, #12
	STMFD	SPI, {R0-R2}
\$125C	BL	R
\$1260	ADD	SP, SP, #12
\$1264		
\$1268	MOV	SP, FP
	LDMFD	SPI, {FP, PC}

```

void R( int I, int J, int *O )
{
  int A, B, C;
  *O = I+J;
  if( A == 0 )
  R( A, B, &C );
}

```


Allocazione dinamica della memoria in ARM

	ADDR	DATA	
	\$FF9C	5	
	\$FFA0	21	
	\$FFA4	12	
	\$FFA8	\$FFC8	FP"
	\$FFAC	\$1260	LR(PC)
SP	\$FFA8	\$FFB0	&C
		\$FFB4	7 B
		\$FFB8	0 A
FP	\$FFA8	\$FFBC	7 C
		\$FFC0	7 B
		\$FFC4	0 A
PC	\$1268	\$FFC8	\$FFEC FP'
		\$FFCC	\$1154 LR(PC)
		\$FFD0	\$FFDC &Z
LR	\$1260	\$FFD4	27 Y
		\$FFD8	111 X
		\$FFDC	138 Z
		\$FFE0	27 Y
		\$FFE4	111 X
		\$FFE8	12 W
		\$FFEC	\$FFF8
		\$FFF0	\$400
		\$FFF4	
		\$FFF8	

La condizione A==0 è falsa: non viene eseguita la chiamata ricorsiva alla subroutine R
Vengono eseguite le istruzioni successive fino al termine della funzione R quindi esco da R.
Ripristino il vecchio SP leggendolo da FP: ho de-allocato lo spazio delle variabili locali

\$1200	STMFD	SPI, {FP,LR}	void R(int I, int J, int *O)
	MOV	FP, SP	{
	SUB	SP, SP, #12	int A, B, C;
\$120C			*O = I+J;
	LDR	R0, [FP, #16]	if(A == 0)
	LDR	R1, [FP, #12]	R(A, B, &C);
	ADD	R0, R0, R1	}
	LDR	R1, [FP, #8]	
	STR	R0, [R1]	
\$1238			
	LDR	R0, [FP, #-4]	
	CMP	R0, #0	
\$1248	BNE	\$1268	
\$124C	LDR	R2, [FP, #-4]	
	LDR	R1, [FP, #-8]	
	SUB	R0, FP, #12	
	STMFD	SPI, {R0-R2}	
\$125C	BL	R	
\$1260	ADD	SP, SP, #12	
\$1264			
\$1268	MOV	SP, FP	
	LDMFD	SPI, {FP, PC}	

Allocazione dinamica della memoria in ARM

	ADDR	DATA	
	\$FF9C	5	
	\$FFA0	21	
	\$FFA4	12	
	\$FFA8	\$FFC8	
	\$FFAC	\$1260	
SP	\$FFB0	\$FFBC	&C
	\$FFB4	7	B
	\$FFB8	0	A
FP	\$FFBC	7	C
	\$FFC0	7	B
	\$FFC4	0	A
PC	\$FFC8	\$FFEC	FP'
	\$FFCC	\$1154	LR(PC)
	\$FFD0	\$FFDC	&Z
LR	\$FFD4	27	Y
	\$FFD8	111	X
	\$FFDC	138	Z
	\$FFE0	27	Y
	\$FFE4	111	X
	\$FFE8	12	W
	\$FFEC	\$FFF8	
	\$FFF0	\$400	
	\$FFF4		
	\$FFF8		

Ripristino il vecchio FP leggendolo dallo stack
Ho completato la de-allocazione del frame della (seconda) chiamata a R.
Contemporaneamente, ho aggiornato il PC, inserendo il valore che avevo salvato da LR

\$1200	STMFD	SPI, {FP,LR}	void R(int I, int J, int *O)
	MOV	FP, SP	{
	SUB	SP, SP, #12	int A, B, C;
\$120C			*O = I+J;
	LDR	R0, [FP, #16]	if(A == 0)
	LDR	R1, [FP, #12]	R(A, B, &C);
	ADD	R0, R0, R1	}
	LDR	R1, [FP, #8]	
	STR	R0, [R1]	
\$1238			
	LDR	R0, [FP, #-4]	
	CMP	R0, #0	
\$1248	BNE	\$1268	
\$124C	LDR	R2, [FP, #-4]	
	LDR	R1, [FP, #-8]	
	SUB	R0, FP, #12	
	STMFD	SPI, {R0-R2}	
\$125C	BL	R	
\$1260	ADD	SP, SP, #12	
\$1264			
\$1268	MOV	SP, FP	
	LDMFD	SPI, {FP, PC}	

Allocazione dinamica della memoria in ARM

	ADDR	DATA	
	\$FF9C	5	
	\$FFA0	21	
	\$FFA4	12	
	\$FFA8	\$FFC8	
	\$FFAC	\$1260	
SP	\$FFBC	\$FFB0	\$FFBC
		\$FFB4	7
		\$FFB8	0
FP	\$FFC8	\$FFBC	7
		\$FFC0	7
		\$FFC4	0
PC	\$1264	\$FFC8	\$FFEC
		\$FFCC	\$1154
		\$FFD0	\$FFDC
LR	\$1260	\$FFD4	27
		\$FFD8	111
		\$FFDC	138
		\$FFE0	27
		\$FFE4	111
		\$FFE8	12
		\$FFEC	\$FFF8
		\$FFF0	\$400
		\$FFF4	
		\$FFF8	

Sono tornato alla funzione chiamante: la prima chiamata di R.
Tornato dalla funzione, libero la memoria usata per il passaggio dei parametri:
incrementando SP.

```

$1200 STMFD   SP!, {FP,LR}
      MOV     FP, SP
      SUB     SP, SP, #12

$120C   LDR     R0, [FP, #16]
      LDR     R1, [FP, #12]
      ADD     R0, R0, R1
      LDR     R1, [FP, #8]
      STR     R0, [R1]

$1238   LDR     R0, [FP, #-4]
      CMP     R0, #0
$1248   BNE     $1268

$124C   LDR     R2, [FP, #-4]
      LDR     R1, [FP, #-8]
      SUB     R0, FP, #12
      STMFD   SP!, {R0-R2}
$125C   BL      R
$1260   ADD     SP, SP, #12
$1264
$1268   MOV     SP, FP
      LDMFD   SP!, {FP, PC}
  
```

```

void R( int I, int J, int *O )
{
  int A, B, C;
  *O = I+J;
  if( A == 0 )
    R( A, B, &C );
}
  
```

Allocazione dinamica della memoria in ARM

	ADDR	DATA	
	\$FF9C	5	
	\$FFA0	21	
	\$FFA4	12	
	\$FFA8	\$FFC8	
	\$FFAC	\$1260	
SP	\$FFC8	\$FFB0	\$FFBC
		\$FFB4	7
		\$FFB8	0
FP	\$FFC8	\$FFBC	7
		\$FFC0	7
		\$FFC4	0
PC	\$1268	\$FFC8	\$FFEC
		\$FFCC	\$1154
		\$FFD0	\$FFDC
LR	\$1260	\$FFD4	27
		\$FFD8	111
		\$FFDC	138
		\$FFE0	27
		\$FFE4	111
		\$FFE8	12
		\$FFEC	\$FFF8
		\$FFF0	\$400
		\$FFF4	
		\$FFF8	

FP'
LR(PC)

&Z

Y

X

Z

Y

X

W

La funzione R riprende la sua esecuzione.

Il suo frame è uguale al momento in cui ha chiamato R, ad eccezione della variabile C che è stata modificata dalla subroutine, poichè era stata passata per indirizzo.

Al termine delle istruzioni di R, chiudo la funzione: ripristino il vecchio SP leggendolo da FP.

```

$1200 STMFD   SP!, {FP,LR}
      MOV     FP, SP
      SUB     SP, SP, #12

$120C   LDR     R0, [FP, #16]
      LDR     R1, [FP, #12]
      ADD     R0, R0, R1
      LDR     R1, [FP, #8]
      STR     R0, [R1]

$1238   LDR     R0, [FP, #-4]
      CMP     R0, #0
      BNE     $1248

$1248   LDR     R2, [FP, #-4]
      LDR     R1, [FP, #-8]
      SUB     R0, FP, #12
      STMFD   SP!, {R0-R2}

$125C   BL      R
$1260   ADD     SP, SP, #12
$1264
$1268   MOV     SP, FP
      LDMFD   SP!, {FP, PC}
  
```

```

void R( int I, int J, int *O )
{
  int A, B, C;
  *O = I+J;
  if( A == 0 )
    R( A, B, &C );
}
  
```

Allocazione dinamica della memoria in ARM

	ADDR	DATA	
	\$FF9C	5	
	\$FFA0	21	
	\$FFA4	12	
	\$FFA8	\$FFC8	
	\$FFAC	\$1260	
SP	\$FFD0	\$FFB0	\$FFB0
		\$FFB4	7
		\$FFB8	0
FP	\$FFEC	\$FFBC	7
		\$FFC0	7
		\$FFC4	0
PC	\$1154	\$FFC8	\$FFEC
		\$FFCC	\$1154
		\$FFD0	\$FFDC
LR	\$1260	\$FFD4	27
		\$FFD8	111
		\$FFDC	138
		\$FFE0	27
		\$FFE4	111
		\$FFE8	12
		\$FFEC	\$FFF8
		\$FFF0	\$400
		\$FFF4	
		\$FFF8	

&Z
Y
X
Z
Y
X
W

Ripristino FP leggendolo dallo stack

Ho terminato la de-allocazione del frame di R.

Contemporaneamente, ho aggiornato il PC, inserendo il valore che avevo salvato da LR

```

$1200 STMFD   SP!, {FP,LR}
      MOV     FP, SP
      SUB     SP, SP, #12

$120C   LDR     R0, [FP, #16]
      LDR     R1, [FP, #12]
      ADD     R0, R0, R1
      LDR     R1, [FP, #8]
      STR     R0, [R1]

$1238   LDR     R0, [FP, #-4]
      CMP     R0, #0
$1248   BNE     $1268

$124C   LDR     R2, [FP, #-4]
      LDR     R1, [FP, #-8]
      SUB     R0, FP, #12
      STMFD   SP!, {R0-R2}
$125C   BL      R
$1260   ADD     SP, SP, #12
$1264
$1268   MOV     SP, FP
      LDMFD   SP!, {FP, PC}
  
```

```

void R( int I, int J, int *O )
{
  int A, B, C;
  *O = I+J;
  if( A == 0 )
    R( A, B, &C );
}
  
```

Allocazione dinamica della memoria in ARM

		ADDR	DATA	
		\$FF9C	5	
		\$FFA0	21	
		\$FFA4	12	
		\$FFA8	\$FFC8	
		\$FFAC	\$1260	
SP	\$FFDC	\$FFB0	\$FFBC	
		\$FFB4	7	
		\$FFB8	0	
FP	\$FFEC	\$FFBC	7	
		\$FFC0	7	
		\$FFC4	0	
PC	\$1158	\$FFC8	\$FFEC	
		\$FFCC	\$1154	
		\$FFD0	\$FFDC	
LR	\$1260	\$FFD4	27	
		\$FFD8	111	
		\$FFDC	138	Z
		\$FFE0	27	Y
		\$FFE4	111	X
		\$FFE8	12	W
		\$FFEC	\$FFF8	
		\$FFF0	\$400	
		\$FFF4		
		\$FFF8		

Sono tornato alla funzione chiamante, MAIN.
Tornato dalla funzione, libero la memoria usata per il passaggio dei parametri incrementando SP.

\$1100	STMFD	SP!, {FP,LR}	int main()
	MOV	FP, SP	{
	SUB	SP, SP, #16	int W, X, Y, Z;
\$110C			R(X, Y, &Z);
			}
\$1140	LDR	R2, [FP, #-8]	
	LDR	R1, [FP, #-12]	
	SUB	R0, FP, #16	
	STMFD	SP!, {R0-R2}	
\$1150	BL	R	
\$1154	ADD	SP, SP, #12	
\$1158			
	MOV	SP, FP	
	LDMFD	SP!, {FP, PC}	

	ADDR	DATA
	\$FF9C	5
	\$FFA0	21
	\$FFA4	12
	\$FFA8	\$FFC8
	\$FFAC	\$1260
SP	\$FFF4	\$FFB0
	\$FFB4	7
	\$FFB8	0
FP	\$FFF8	\$FFBC
	\$FFC0	7
	\$FFC4	0
PC	\$400	\$FFC8
	\$FFCC	\$1154
	\$FFD0	\$FFDC
LR	\$1260	\$FFD4
	\$FFD8	27
	\$FFDC	111
	\$FFE0	138
	\$FFE4	27
	\$FFE8	111
	\$FFE8	12
	\$FFEC	\$FFF8
	\$FFF0	\$400
	\$FFF4	
	\$FFF8	

Proseguo l'esecuzione della funzione MAIN
Al termine delle istruzioni, chiudo la funzione: de-alloco il frame, ripristino il FP precedente e, grazie al load multiplo, contemporaneamente aggiorno il PC.

\$1100	STMFD	SP!, {FP,LR}	int main()
	MOV	FP, SP	{
	SUB	SP, SP, #16	int W, X, Y, Z;
\$110C			R(X, Y, &Z);
			}
\$1140	LDR	R2, [FP, #-8]	
	LDR	R1, [FP, #-12]	
	SUB	R0, FP, #16	
	STMFD	SP!, {R0-R2}	
\$1150	BL	R	
\$1154	ADD	SP, SP, #12	
\$1158			
	MOV	SP, FP	
	LDMFD	SP!, {FP, PC}	