Assembly Program Example

	Memory address label	Operation	Addressing or data information
Assembler directive		ORIGIN	100
Statements that generate machine instructions		LD	R2, N
		CLR	R3
		MOV	R4, #NUM1
	LOOP:	LD	R5, (R4)
		ADD	R3, R3, R5
		ADD	R4, R4, #4
		SUB	R2, R2, #1
		BGT	R2, R0, LOOP
		ST	R3, SUM
		next instruction	
Assembler directives		ORIGIN	200
	SUM:	RESERVE	4
	N:	DATAWORD	150
	NUM1:	RESERVE	600
		END	

Pseudo to ARM/Thumb

```
        LD      R2, N
        CLR     R3
        MOV     R4, #NUM1
LOOP:   LD      R5, (R4)
        ADD     R3, R3, R5
        ADD     R4, R4, #4
        SUB     R2, R2, #1
        BGT     R2, R0, LOOP
        ST      R3, SUM

SUM:    RESERVE 4
N:      DATAWORD 150
NUM1:   RESERVE 600
```

36 ARM Thumb Instructions

ADC	Add with Carry
ADD	Add
AND	AND
ASR	Arithmetic Shift Right
B	Unconditional branch
B*xx*	Conditional branch
BIC	Bit Clear
BL	Branch and Link
BX	Branch and Exchange
CMN	Compare Negative
CMP	Compare
EOR	EOR

LDMIA	Load multiple
LDR	Load word
LDRB	Load byte
LDRH	Load halfword
LSL	Logical Shift Left
LDSB	Load sign-extended byte
LDSH	Load sign-extended halfword
LSR	Logical Shift Right
MOV	Move register
MUL	Multiply
MVN	Move Negative register

NEG	Negate
ORR	OR
POP	Pop registers
PUSH	Push registers
ROR	Rotate Right
SBC	Subtract with Carry
STMIA	Store Multiple
STR	Store word
STRB	Store byte
STRH	Store halfword
SWI	Software Interrupt
SUB	Subtract
TST	Test bits

Ri ← #offset8 or Ri ← Rj

Pseudo to ARM/Thumb

```
        LD      R2, N
        CLR     R3
        MOV     R4, #NUM1
LOOP:   LD      R5, (R4)
        ADD     R3, R3, R5
        ADD     R4, R4, #4
        SUB     R2, R2, #1
        BGT     R2, R0, LOOP
        ST      R3, SUM
```

```
        LDR     R1, N           Load count into R1.
        LDR     R2, =NUM1       Load address NUM1 into R2.
        MOV     R0, #0          Clear accumulator R0.
LOOP    LDR     R3, [R2], #4    Load next number into R3.
        ADD     R0, R0, R3      Add number into R0.
        SUBS    R1, R1, #1      Decrement loop counter R1.
        BGT     LOOP            Branch back if not done.
        STR     R0, SUM         Store sum.
```

```
SUM:    RESERVE     4
N:      DATAWORD    150
NUM1:   RESERVE     600
```

ARM Pseudo Instructions

- The ARM assembler supports a number of pseudo-instructions that are translated into the appropriate instructions at assembly time.
- ARM Manual Pseudo Instructions here:
- https://developer.arm.com/documentation/dui0802/b/A32-and-T32-Instructions/Pseudo-instructions?lang=en

LDR Pseudo-instruction Syntax

LDR pseudo-instruction

Load a register with either:

- A 32-bit immediate value.
- An address.

Note

This description is for the `LDR` pseudo-instruction only, and not the `LDR` instruction.

Syntax

```
LDR{cond}{.W} Rt, =expr
```

```
LDR{cond}{.W} Rt, =label_expr
```

LDR Pseudo-instruction Examples

- LDR r3,=0xff0 ; loads 0xff0 into R3
  ; => MOV.W r3,#0xff0

- LDR r2,=place ; loads the address of
  ; place into R2

How about NOP? MOV r8, r8