Punctuation	Meaning
;	End of statement
:	Defines a label
,	Separates elements of a list
( )	Start and end of a parameter list
{ }	Start and stop of a compound statement
[ ]	Start and stop of a array index
" "	Start and stop of a string
' '	Start and stop of a character constant

Data type	Range	Precision
unsigned char	0 to +255	8-bit unsigned
signed char	-128 to +127	8-bit signed
unsigned int	32-bit in Keil	compiler-dependent
int	32-bit in Keil	compiler-dependent
unsigned short	0 to +65535	16-bit unsigned
short	-32768 to +32767	16-bit signed
unsigned long	0 to 4294967295L	32-bit unsigned
long	-2147483648L to 2147483647L	32-bit signed
float	$\pm10^{-38}$ to $\pm10^{+38}$	32-bit float
double	$\pm10^{-308}$ to $\pm10^{+308}$	64-bit float

Hex Digit	Decimal	Binary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A or a	10	1010
B or b	11	1011
C or c	12	1100
D or d	13	1101
E or e	14	1110
F or f	15	1111

Operation	Meaning
=	Assignment statement
?	Selection
<	Less than
>	Greater than
!	Logical not (T to F, F to T)
~	1's complement
+	Addition
-	Subtraction
*	Multiply or pointer reference
/	Divide
%	Modulo, division remainder
|	Logical or
&	Logical and, or address of
^	Logical exclusive or
.	Used to access parts of a struct

Operation	Meaning
==	Equal to comparison
<=	Less than or equal to
>=	Greater than or equal to
!=	Not equal to
<<	Shift left
>>	Shift right
++	Increment
--	Decrement
&&	Boolean and
||	Boolean or
+=	Add value to
-=	Subtract value to
*=	Multiply value to
/=	Divide value to
|=	Or value to
&=	And value to
^=	Exclusive or value to
<<=	Shift value left
>>=	Shift value right
%=	Modulo divide value to
->	Pointer to a structure

Fundamental Boolean Laws	
A & B = B & A	Commutative Law
A | B = B | A	Commutative Law
(A & B) & C = A & (B & C)	Associative Law
(A | B) | C = A | (B | C)	Associative Law
(A | B) & C = (A & C) | (B & C)	Distributive Law
(A & B) | C = (A | C) & (B | C)	Distributive Law
A & 0 = 0	Identity of 0
A | 0 = A	Identity of 0
A & 1 = A	Identity of 1
A | 1 = 1	Identity of 1
A | A = A	Property of OR
A | (~A) = 1	Property of OR
A & A = A	Property of AND
A & (~A) = 0	Property of AND
~(~A) = A	Inverse
~(A | B) = (~A) & (~B)	De Morgan's Theorem
~(A & B) = (~A) | (~B)	De Morgan's Theorem

Common Register Operations
register |= (1<<bit);   // set bit 0 to 31
register &= ~(1<<bit);   // clear bit 0 to 31
data = register & (1<<bit);   // "isolate" bit 0 to 31
register ^= (1<<bit);   // complement bit 0 to 31

Keyword	Meaning
__asm	Specify a function is written in assembly code (specific to ARM Keil™uVision®)
auto	Specifies a variable as automatic (created on the stack)
break	Causes the program control structure to finish
case	One possibility within a switch statement
char	Defines a number with a precision of 8 bits
const	Defines parameter as constant in ROM, and defines a local parameter as fixed value
continue	Causes the program to go to beginning of loop
default	Used in switch statement for all other cases
do	Used for creating program loops
double	Specifies variable as double precision floating point
else	Alternative part of a conditional
extern	Defined in another module
float	Specifies variable as single precision floating point
for	Used for creating program loops
goto	Causes program to jump to specified location
if	Conditional control structure
int	Defines a number with a precision that will vary from compiler to compiler
long	Defines a number with a precision of 32 bits
register	Specifies how to implement a local
return	Leave function
short	Defines a number with a precision of 16 bits
signed	Specifies variable as signed (default)
sizeof	Built-in function returns the size of an object
static	Stored permanently in memory, accessed locally
struct	Used for creating data structures
switch	Complex conditional control structure
typedef	Used to create new data types
unsigned	Always greater than or equal to zero
void	Used in parameter list to mean no parameter
volatile	Can change implicitly outside the direct action of the software.
while	Used for creating program loops