1) Convert the following numbers to 16-bit 2's-complement binary numbers.

I have added some extra results, that were not called for, these are done to show other forms of these numbers.

A) 1024_{10}

1024/2 = 512.0 => 0
512/2 = 256.0 => 0
256/2 = 128.0 => 0
128/2 = 64.0 => 0
64/2 = 32.0 => 0
32/2 = 16.0 => 0
16/2 = 8.0 => 0
8/2 = 4.0 => 0
4/2 = 2.0 => 0
2/2 = 1.0 => 0
1/2 = 0.5 => 1

$0000\ 0100\ 0000\ 0000_2$ *Extra 0400_{16} or 0x0400;*

B) -5555_{10} To do a negative number we will first convert the magnitude of the number then convert that to negative with a two's complement operation.

5555/2 = 2777.5 => 1
2777/2 = 1388.5 => 1
1388/2 = 694.0 => 0
694/2 = 347.0 => 0
347/2 = 173.5 => 1
173/2 = 86.5 => 1
86/2 = 43.0 => 0
43/2 = 21.5 => 1
21/2 = 10.5 => 1
10/2 = 5.0 => 0
5/2 = 2.5 => 1
2/2 = 1.0 => 0
1/2 = 0.5 => 1

$0001\ 0101\ 1011\ 0011_2 = 5555_{10}$
$1110\ 1010\ 0100\ 1100$
$+\ 0000\ 0000\ 0000\ 0001$
$1110\ 1010\ 0100\ 1101 = -5555_{10}$ *Extra $EA4D_{16}$ or 0xEA4D;*

C) -1_{10} Again the magnitude is written out first, then 2's complement operation.

1/2 = 0.5 => 1

$0000\ 0000\ 0000\ 0001_2 = 1_{10}$
$1111\ 1111\ 1111\ 1110$
$+\ 0000\ 0000\ 0000\ 0001$
$1111\ 1111\ 1111\ 1111 = -1_{10}$ *Extra $FFFF_{16}$ or 0xFFFF;*

D) 32000_{10}

Integer Part	/ 2	Bit
32000	16000	"=> 0"
16000	8000	"=> 0"
8000	4000	"=> 0"
4000	2000	"=> 0"
2000	1000	"=> 0"
1000	500	"=> 0"
500	250	"=> 0"
250	125	"=> 0"
125	62.5	"=> 1"
62	31	"=> 0"
31	15.5	"=> 1"
15	7.5	"=> 1"
7	3.5	"=> 1"
3	1.5	"=> 1"
1	0.5	"=> 1"
0	0	"=> 0"

$0111\ 1101\ 0000\ 0000_2 = 32000_{10}$ *Extra* $7D00_{16}$ *or 0x7D00;*

E) $2AC9_{16}$ In the case of hexadecimal we simply change each digit into it binary equivalent.

2	A	C	9	
0010	1010	1100	1001	*Extra 10953*

2) Complete the following c program operations

Note in c all numbers are decimal, unless specifically shown otherwise.

A) 0x032 & 32;

$$\begin{array}{l} 0000\ 0000\ 0011\ 0010 \\ \underline{\&\ 0000\ 0000\ 0010\ 0000} \\ 0000\ 0000\ 0010\ 0000_2 = 32 \end{array}$$

Note all bits are lost except for where the mask is a one.

B) 104 + 0x25; $104 + (0010\ 0101) = 104 + (32 + 4 + 1) = 104 + 37 = 141$

C) 255 ^ 0x20;

$$\begin{array}{l} 0000\ 0000\ 1111\ 1111 \\ \underline{\wedge\ 0000\ 0000\ 0010\ 0000} \\ 0000\ 0000\ 1101\ 1111_2 = 223 \end{array}$$

Note all bits are preserved except for where the mask is a one, and that bit is inverted.

D) 0x33 | 0x0c; 0000 0000 0011 0011
| 0000 0000 0000 1100
0000 0000 0011 1111_2 = 63 *Note all bits are preserved except for where the mask is a one, and those bit are forced high.*

E) 125 & ~0x18; First consider ~0x18
which becomes ~(0000 0000 0001 1000)
or 1111 1111 1110 0111

0000 0000 0111 1101
& 1111 1111 1110 0111
0000 0000 0110 0101_2 = 101
Note all bits are preserved except for where the mask is a one, and those bit are forced low.