### The University of Texas at Arlington

### Lecture 6 PIC Programming in C





CSE 3442/5442

### **Embedded Systems 1**

Based heavily on slides by Dr. Gergely Záruba and Dr. Roger Walker



# **Code Space Limitations**

- On a general purpose PC, we don't usually care about our program's size
- MB/GB/TB range for general purpose PCs
   Ex: 1300 line .C file 50 KB → 40 KB .hex file
- 2MB max in PIC18's Program ROM
- For our PIC18F452 → Only 32KB

- See datasheet



# Why C over ASM?

- While Assembly Language produces a much smaller .HEX file than C...
  - More human-readable in C
    - Easier to write and less time consuming
  - C is easier to modify and update
    - Don't care about absolute ROM locations
  - Access to many C function libraries
  - C code is portable and can be used on other microcontrollers with little or no modification



### C Integer Data Types (Generic)

| Туре   | Explanation   | Format Specifier                     |
|--|---|--------------------------------------|
| char   | Smallest addressable unit of the machine that can contain basic character set. It is an integer type. Actual type can be either signed or unsigned depending on the implementation. It contains CHAR_BIT bits. <sup>[3]</sup>   | %с                                   |
| signed char  | Of the same size as char, but guaranteed to be signed. Capable of containing <b>at least</b> the [-127, +127] range; <sup>[3][4]</sup>  | %C (or %hhi for numerical<br>output) |
| unsigned char  | Of the same size as char, but guaranteed to be unsigned. It is represented in binary notation without padding bits; thus, its range is exactly [0, 2 CHAR_BIT - 1]. <sup>[5]</sup>  | %C (or %hhu for numerical<br>output) |
| short<br>short int<br>signed short<br>signed short int                 | Short signed integer type. Capable of containing <b>at least</b> the [-32767, +32767] range; <sup>[3][4]</sup> thus, it is at least 16 bits in size. The negative value is -32767 (not -32768) due to the one's-complement and sign-magnitude representations allowed by the standard, though the two's-complement representation is much more common. <sup>[6]</sup> | %hi                                  |
| unsigned short<br>unsigned short int                                   | Similar to short, but unsigned.   | %hu                                  |
| int<br>signed<br>signed int  | Basic signed integer type. Capable of containing <b>at least</b> the [-32767, +32767] range; <sup>[3][4]</sup> thus, it is at least 16 bits in size.  | %i or %d                             |
| unsigned<br>unsigned int   | Similar to int, but unsigned.   | %u                                   |
| long<br>long int<br>signed long<br>signed long int                     | Long signed integer type. Capable of containing <b>at least</b> the [-2147483647, +2147483647] range; <sup>[3][4]</sup> thus, it is at least 32 bits in size.   | %li                                  |
| unsigned long<br>unsigned long int                                     | Similar to long, but unsigned.  | %lu                                  |
| long long<br>long long int<br>signed long long<br>signed long long int | Long long signed integer type. Capable of containing <b>at least</b> the [-9223372036854775807, +9223372036854775807] range; <sup>[3][4]</sup> thus, it is at least 64 bits in size. Specified since the C99 version of the standard.   | %Ili                                 |
| unsigned long long<br>unsigned long long int                           | Similar to long long, but unsigned. Specified since the C99 version of the standard.  | %llu                                 |



### C Integer Data Types (C18 Compiler)

#### TABLE 2-1: INTEGER DATA TYPE SIZES AND LIMITS

| Туре                  | Size    | Minimum        | Maximum       |
|-----------------------|---------|----------------|---------------|
| char <sup>(1,2)</sup> | 8 bits  | -128           | 127           |
| signed char           | 8 bits  | -128           | 127           |
| unsigned char         | 8 bits  | 0              | 255           |
| int                   | 16 bits | -32,768        | 32,767        |
| unsigned int          | 16 bits | 0              | 65,535        |
| short                 | 16 bits | -32,768        | 32,767        |
| unsigned short        | 16 bits | 0              | 65,535        |
| short long            | 24 bits | -8,388,608     | 8,388,607     |
| unsigned short long   | 24 bits | 0              | 16,777,215    |
| long                  | 32 bits | -2,147,483,648 | 2,147,483,647 |
| unsigned long         | 32 bits | 0              | 4,294,967,295 |



### C Integer Data Types (XC8 Compiler)

#### TABLE 5-1: INTEGER DATA TYPES

| Туре                | Size (bits) | Arithmetic Type  |
|---------------------|-------------|------------------|
| bit                 | 1           | Unsigned integer |
| signed char         | 8           | Signed integer   |
| unsigned char       | 8           | Unsigned integer |
| signed short        | 16          | Signed integer   |
| unsigned short      | 16          | Unsigned integer |
| signed int          | 16          | Signed integer   |
| unsigned int        | 16          | Unsigned integer |
| signed short long   | 24          | Signed integer   |
| unsigned short long | 24          | Unsigned integer |
| signed long         | 32          | Signed integer   |
| unsigned long       | 32          | Unsigned integer |
| signed long long    | 32          | Signed integer   |
| unsigned long long  | 32          | Unsigned integer |



### Unsigned char (0 to 255)

- PIC18 is 8-bit architecture, char type (8 bits) is the most natural choice
- C compilers use signed char (-128 to +127) by default unless we put "unsigned"
  - char == signed char

```
Write a C18 program to send values 00-FF to Port B.
Solution:
#include <P18F458.h> //for TRISB and PORTB declarations
void main(void)
{
    unsigned char z;
    TRISB = 0; //make Port B an output
    for(z=0;z<=255;z++)
        PORTB = z;
    while(1); //NEEDED IF RUNNING IN HARDWARE
}</pre>
```



| Write a C18 program to send hex values for ASCII characters of 0, 1, 2, 3, 4, 5, A, B,       0x21 33 t       0x41 66 A         0x22 34 t       0x42 66 B         0x23 35 t       0x44 66 D         0x24 36 S       0x44 71 B         0x27 39 t       0x47 71 B         0x28 40 (0000000000000000000000000000000000   |  | Hex  | Dec  | Char   | Hex  | Dec ( | Char   |
|--|--|------|------|--------|------|-------|--------|
| Write a C18 program to send hex values for ASCII characters of 0, 1, 2, 3, 4, 5, A, B,       0x22 33 ± 0       0x41 65 A         C, and D to Port B.       0x23 35 ± 0       0x43 67 C         Solution:       0x27 39 ± 0       0x45 69 E         #include <p18f458.h>       0x44 68 D       0x23 35 ± 0         void main (void)       0x24 36 ± 0       0x44 72 H         {       0x24 36 ± 0       0x44 73 H         unsigned char mynum[] = "012345ABCD";//data is stored in RAM       0x22 4 ± 0       0x48 75 K         unsigned char z;       //make Port B an output       0x25 40 ± 0       0x50 80 P         for (z=0; z&lt;10; z++)</p18f458.h>  |  | 0x20 | 32   | Space  | 0x40 | 64    | 6      |
| Write a C18 program to send hex values for ASCII characters of 0, 1, 2, 3, 4, 5, A, B,       0x22 33 # " 0x43 66 B         C, and D to Port B.       0x26 38 6 0x46 70 F         Solution:       0x27 39 ' 0x47 71 6         #include <p18f458.h>       0x24 2 * 0x44 74 J         void main(void)       0x22 44 . 0x48 75 K         {       0x22 44 . 0x48 77 M         unsigned char z;       0x24 5 . 0x49 77 M         TRISE = 0;       //make Port B an output         for (z=0;z&lt;10;z++)</p18f458.h>  |  | 0x21 | . 33 | 1      | 0x41 | 65    | А      |
| Write a C18 program to send hex values for ASCII characters of 0, 1, 2, 3, 4, 5, A, B,<br>C, and D to Port B.<br>Solution:<br>#include <p18f458.h><br/>void main(void)<br/>{<br/>unsigned char mynum[] = "012345ABCD";//data is stored in RAM<br/>unsigned char z;<br/>TRISB = 0; //make Port B an output<br/>for(z=0;z&lt;10;z++)<br/>PORTB = mynum[z];<br/>while(1); //stay here forever<br/>}<br/>dxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</p18f458.h>   |  | 0x22 | 34   | "      | 0x42 | 66    | в      |
| Write a C18 program to send hex values for ASCII characters of 0, 1, 2, 3, 4, 5, A, B,<br>C, and D to Port B.<br>Solution:<br>#include <p18f458.h><br/>void main(void)<br/>{<br/>unsigned char mynum[] = "012345ABCD";//data is stored in RAM<br/>unsigned char z;<br/>TRISB = 0; //make Port B an output<br/>for(z=0;z&lt;10;z++)<br/>PORTB = mynum[z];<br/>while(1); //stay here forever<br/>}<br/>(0x28 40 (0x48 70 F<br/>0x27 39 (0x47 71 G<br/>0x28 40 (0x48 72 B<br/>0x28 40 (0x48 72 B<br/>0x28 41 ) 0x49 73 I<br/>0x28 42 * 0x48 74 J<br/>0x28 42 * 0x48 74 J<br/>0x28 40 (0x48 77 H<br/>0x28 40 (0x48 78 N<br/>0x28 40 (0x58 80 P<br/>0x33 51 3 0x53 83 S<br/>0x34 52 4 0x58 88 x<br/>0x35 55 7 0x57 87 W<br/>0x38 56 8 0x58 88 x<br/>0x38 56 8 0x58 88 x<br/>0x38 59 (0x58 99 Y<br/>0x38 58 (0x58 98 Y<br/>0x38 56 (0x56 88 0x58 98 Y<br/>0x38 56 (0x56 98 0x58 99 1<br/>0x38 62 (0x56 92 \<br/>0x38 63 7 0x57 99 y55<br/>(0x58 94 7<br/>(0x58 94 7<br/>(0x58 94 7<br/>(0x58 94 7<br/>(0x58 94 7<br/>(0x58 94 7)<br/>(0x58 94 7<br/>(0x58 94 7<br/>(0x58 94 7)<br/>(0x58 94</p18f458.h>   |  | 0x23 | 35   | #      | 0x43 | 67    | C      |
| C, and D to Port B.<br>Solution:<br>#include <p18f458.h><br/>void main(void)<br/>{<br/>unsigned char z;<br/>TRISB = 0;<br/>TRISB = 0;<br/>PORTB = mynum[z];<br/>while(1);<br/>}<br/>//stay here forever<br/>}<br/>(<br/>unsigned char z;<br/>TRISB = 0;<br/>C, and D to Port B an output<br/>for (z=0;z&lt;10;z++)<br/>PORTB = mynum[z];<br/>while(1);<br/>}<br/>//stay here forever<br/>}<br/>(<br/>unsigned char z;<br/>TRISB = 0;<br/>C, and D to Port B an output<br/>(<br/>unsigned char z;<br/>TRISB = 0;<br/>C, and D to Port B an output<br/>(<br/>unsigned char z;<br/>C, and D to Port B an output<br/>(<br/>unsigned char z;<br/>C, and D to Port B an output<br/>(<br/>unsigned char z;<br/>C, and D to Port B an output<br/>(<br/>unsigned char z;<br/>C, and D to Port B an output<br/>(<br/>unsigned char z;<br/>C, and D to Port B an output<br/>(<br/>unsigned char z;<br/>(<br/>unsigned char z;<br/>(<br/>unsig</p18f458.h> | Write a C18 program to send hex values for ASCII characters of 0 1 2 3 4 5 A B   | 0x24 | 30   | Ş      | 0x44 | 68    | D      |
| C, and D to Port B.<br>Solution:<br>#include <p18f458.h><br/>void main(void)<br/>{<br/>unsigned char z;<br/>TRISB = 0; //make Port B an output<br/>for(z=0;z&lt;10;z++)<br/>PORTB = mynum[z];<br/>while(1); //stay here forever<br/>}<br/>(2020 + 1) (2020 +</p18f458.h>   | The a cro program to some new values for resent characters of 0, 1, 2, 5, 4, 5, 11, D,   | 0x25 | 38   | б<br>2 | 0x45 | 70    | F      |
| Solution:       #include <p18f458.h>       0x48 72 H         woid main(void)       0x28 40 (0x48 72 H)         {       0x28 41 )       0x49 73 I         unsigned char mynum[] = "012345ABCD";//data is stored in RAM       0x22 44 :       0x48 76 K         unsigned char z;       0x20 45 -       0x40 77 M         TRISB = 0;       //make Port B an output       0x30 48 0       0x50 80 P         for(z=0;z&lt;10;z++)</p18f458.h>   | C, and D to Port B.  | 0x27 | 39   | 1      | 0x47 | 71    | G      |
| Solution:       #include <p18f458.h>       0x49 73 I         yoid main(void)       0x28 42 * 0x44 74 J         {       unsigned char mynum[] = "012345ABCD";//data is stored in RAM       0x22 44 , 0x42 76 L         unsigned char z;       0x30 48 0       0x26 77 M         TRISB = 0;       //make Port B an output       0x32 50 2       0x28 83 S         for(z=0;z&lt;10;z++)</p18f458.h>   |  | 0x28 | 40   | (      | 0x48 | 72    | Н      |
| bintion:       #include <p18f458.h>       0x2A 42       *       0x4A 74       J         #include <p18f458.h>       void main (void)       0x2B 43       +       0x4B 75       K         void main (void)       {       0x2D 44       ,       0x4C 76       L         unsigned char z;       .       0x2D 45       -       0x4D 77       M         unsigned char z;       .       .       0x4D 77       M         for (z=0;z&lt;10;z++)</p18f458.h></p18f458.h>   | Solution:  | 0x29 | 41   | j      | 0x49 | 73    | I      |
| <pre>#include <p18f458.h> void main(void) {     unsigned char mynum[] = "012345ABCD";//data is stored in RAM     unsigned char z;     TRISB = 0;</p18f458.h></pre>   | Solution.  | 0x2A | 42   | *      | 0x4A | 74    | J      |
| <pre>void main(void) {     unsigned char mynum[] = "012345ABCD";//data is stored in RAM     unsigned char z;     TRISB = 0;</pre>  | #include <p18f458.h></p18f458.h>   | 0x2B | 43   | +      | 0x4B | 75    | K      |
| <pre>{     (     unsigned char mynum[]= "012345ABCD";//data is stored in RAM     unsigned char z;     TRISB = 0;</pre>   | void main(void)  | 0x2C | 44   | 7      | 0x4C | 76    | L      |
| unsigned char mynum[] = "012345ABCD";//data is stored in RAM       0x2E 46 . 0x4E 78 N         unsigned char z;       0x2F 47 / 0x4F 78 0         TRISB = 0;       //make Port B an output         for(z=0;z<10;z++)   | { Contractions are the contract of the second of the secon | 0x2D | 45   | -      | 0x4D | 77    | М      |
| unsigned char z;       //make Port B an output       0x2F 47 7 0x55 81 0         TRISB = 0;       //make Port B an output       0x31 49 1 0x51 81 0         for(z=0;z<10;z++)  | ungigned abor munum[] = "01224EAPCD", //data is stored in DAM  | 0x2E | 46   | •      | 0x4E | 78    | N      |
| Unsigned char z;<br>TRISB = 0; //make Port B an output<br>for(z=0;z<10;z++)<br>PORTB = mynum[z];<br>while(1); //stay here forever<br>}   | unsigned char mynum[] = 012345ABCD ;//data is stored in RAM  | 0x2F | 4/   | /      | 0x4F | 79    | D      |
| TRISB = 0;       //make Port B an output         for(z=0;z<10;z++)   | unsigned char z;   | 0x30 | 40   | 1      | 0x50 | 81    | 0      |
| <pre>for(z=0;z&lt;10;z++)   PORTB = mynum[z];   while(1); //stay here forever } </pre>   | TRISB = 0; //make Port B an output   | 0x32 | 50   | 2      | 0x52 | 82    | R      |
| PORTB = mynum[z];       //stay here forever       0x34 52 4       0x54 84 T         while(1);       //stay here forever       0x36 54 6       0x56 86 V         }       0x37 55 7       0x57 87 W         0x38 56 8       0x58 88 X         0x39 57 9       0x59 89 Y         0x38 56 8       0x58 91 [         0x30 61 =       0x55 93 ]         0x32 62 >       0x55 93 ]         0x38 63 2       0x58 91 [  | for(z=0;z<10;z++)  | 0x33 | 51   | 3      | 0x53 | 83    | S      |
| while(1);       //stay here forever         0x35 53 5       0x55 85 U         0x36 54 6       0x56 86 V         0x37 55 7       0x57 87 W         0x38 56 8       0x58 88 X         0x39 57 9       0x59 89 Y         0x38 58 :       0x58 90 Z         0x30 61 =       0x50 93 ]         0x3E 62 >       0x55 93 ]         0x3F 63 2       0x5F 93 5  | PORTB - mynum[z].  | 0x34 | 52   | 4      | 0x54 | 84    | т      |
| While (1);       //stay here forever         )       0x36 54 6       0x56 86 V         0x37 55 7       0x57 87 W         0x38 56 8       0x58 88 X         0x39 57 9       0x57 87 V         0x38 58 1       0x58 88 X         0x38 59 7       9 0x59 89 Y         0x32 60 <   | rokib = mynam(2),  | 0x35 | 53   | 5      | 0x55 | 85    | U      |
| )       0x37       55       7       0x57       87       W         0x38       56       8       0x58       88       X         0x39       57       9       0x59       89       Y         0x3A       58       :       0x5A       90       Z         0x3B       59       ;       0x5B       91       [         0x3C       60       <  | while(1); //stay here forever  | 0x36 | 54   | 6      | 0x56 | 86    | V      |
| 0x38       56       8       0x58       88       X         0x39       57       9       0x59       89       Y         0x3A       58       :       0x5A       90       Z         0x3B       59       ;       0x5B       91       [         0x3C       60       <  |  | 0x37 | 55   | 7      | 0x57 | 87    | W      |
| 0x39       57       9       0x59       89       Y         0x3A       58       :       0x5A       90       Z         0x3B       59       ;       0x5B       91       [         0x3C       60       <  |  | 0x38 | 56   | 8      | 0x58 | 88    | X      |
| 0x3A       58       :       0x3A       90       2         0x3B       59       ;       0x5B       91       [         0x3C       60       <  |  | 0x39 | 57   | 9      | 0x59 | 89    | Y      |
| 0x3B       33       i       0x3B       91       [         0x3C       60       <  |  | 0x3A | 50   |        | 0x5A | 90    | Z<br>r |
| $\begin{array}{rrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrr$   |  | 0x30 | 60   | i<br>< | 0x50 | 92    |        |
| 0x3E 62 > 0x5E 94<br>0x3F 63 ? 0x5FQ 95  |  | 0x3D | 61   | =      | 0x5D | 93    | ì      |
| 0x3F 63 ? 0x5F <b>Q</b> 95   |  | 0x3E | 62   | >      | 0x5E | 94    | ~      |
|  |  | 0x3F | 63   | ?      | 0x5F | 3 95  |        |



```
#include <xc.h> //PIC18F452
 3
   // This program sends numeric values (hex/dec/bin)
   // for the ASCII characters of 0-5, A-D to PORTB
 4
 5
   void main (void)
 6
 7 🛛 🔤
       unsigned char myNum[] = "012345ABCD"; //data stored in RAM
 8
       unsigned char z;
10
       TRISB = 0; //PORTB is an OUTPUT on all pins
12
13
       for(z=0; z<10; z++)</pre>
14
           PORTB = myNum[z]; //write ASCII char to PORTB
15
16
```



| 1   | <pre>#include <xc.h> //PIC18F452</xc.h></pre>       |
|-----|---|
| 2   |   |
| 3   | // This program sends numeric values (hex/dec/bin)  |
| 4   | // for the ASCII characters of 0-5, A-D to PORTB    |
| 5   |   |
| 6   | void main(void)                                     |
| 7   | <b>₽ {</b>  |
| 8   | unsigned char myNum[] = "012345ABCD"; //data stored |
| 9   | unsigned char z;                                    |
| .0  |   |
| .1  | TRISB = 0; //PORTB is an OUTPUT on all pins         |
| .2  |   |
| .3  | <b>for(</b> z=0; z<10; z++)                         |
| 4   | ₽ {   |
| 5   | PORTB = myNum[z]; //write ASCII char to PORTB       |
| . 6 | }   |
| -   |   |

z = 0 PORTB = '0' (in code) PORTB = 0x30 = 48 (actual)

**PORTB = 0b 0011 0000 (pins)** 

| Hex  | Dec | Char  | Hex  | Dec | Char |
|------|-----|-------|------|-----|------|
| 0x20 | 32  | Space | 0x40 | 64  | 6    |
| 0x21 | 33  | 1     | 0x41 | 65  | A    |
| 0x22 | 34  |       | 0x42 | 66  | в    |
| 0x23 | 35  | #     | 0x43 | 67  | С    |
| 0x24 | 36  | \$    | 0x44 | 68  | D    |
| 0x25 | 37  | 8     | 0x45 | 69  | Е    |
| 0x26 | 38  | &     | 0x46 | 70  | F    |
| 0x27 | 39  | 1     | 0x47 | 71  | G    |
| 0x28 | 40  | (     | 0x48 | 72  | Н    |
| 0x29 | 41  | )     | 0x49 | 73  | I    |
| 0x2A | 42  | *     | 0x4A | 74  | J    |
| 0x2B | 43  | +     | 0x4B | 75  | K    |
| 0x2C | 44  | 7     | 0x4C | 76  | L    |
| 0x2D | 45  | -     | 0x4D | 77  | М    |
| 0x2E | 46  |       | 0x4E | 78  | N    |
| 0x2F | 47  | /     | 0x4F | 79  | 0    |
| 0x30 | 48  | 0     | 0x50 | 80  | Ρ    |
| 0x31 | 49  | 1     | 0x51 | 81  | Q    |
| 0x32 | 50  | 2     | 0x52 | 82  | R    |
| 0x33 | 51  | 3     | 0x53 | 83  | S    |
| 0x34 | 52  | 4     | 0x54 | 84  | т    |
| 0x35 | 53  | 5     | 0x55 | 85  | U    |
| 0x36 | 54  | 6     | 0x56 | 86  | 10   |
| 0x37 | 55  | 7     | 0x57 | 87  | W    |



#include <xc.h> //PIC18F452 // This program sends numeric values (hex/dec/bin) // for the ASCII characters of 0-5, A-D to PORTB Direction void main (void) (TRISB) unsigned char myNum[] = "012345ABCD"; //data sto unsigned char z; 0 39 10 0 38 TRISB = 0; //PORTB is an OUTPUT on all pins 0 37 13 for(z=0; z<10; z++)</pre> 0 36 140 35 PORTB = myNum[z]; //write ASCII char to PORT. 15 16 0 34 2 PIC18F45

 $\mathbf{z} = \mathbf{0}$ **PORTB** = '0' (in code) PORTB = 0x30 = 48 (actual)

 $PORTB = 0b \ 0011 \ 0000 \ (pins)$ 





| 1   | <pre>#include <xc.h> //PIC18F452</xc.h></pre>                  |
|-----|--|
| 2   |  |
| 3   | // This program sends numeric values (hex/dec/bin)             |
| 4   | // for the ASCII characters of 0-5, A-D to PORTB               |
| 5   |  |
| 6   | void main(void)  |
| 7   | ₽ <b>{</b>   |
| 8   | <pre>unsigned char myNum[] = "012345ABCD"; //data stored</pre> |
| 9   | unsigned char z;   |
| 0_0 |  |
| .1  | TRISB = 0; //PORTB is an OUTPUT on all pins                    |
| 2   |  |
| .3  | for(z=0; z<10; z++)  |
| _4  |  |
| 5   | PORTB = myNum[z]; //write ASCII char to PORTB                  |
| - 6 | }  |
|     |  |

z = 1 PORTB = '1' (in code) PORTB = 0x31 = 49 (actual)

**PORTB = 0b 0011 0001 (pins)** 

| Hex  | Dec | Char  | Hex  | Dec | Char              |
|------|-----|-------|------|-----|-------------------|
| 0x20 | 32  | Space | 0x40 | 64  | 6                 |
| 0x21 | 33  | 1     | 0x41 | 65  | А                 |
| 0x22 | 34  |       | 0x42 | 66  | в                 |
| 0x23 | 35  | #     | 0x43 | 67  | С                 |
| 0x24 | 36  | \$    | 0x44 | 68  | D                 |
| 0x25 | 37  | 8     | 0x45 | 69  | Е                 |
| 0x26 | 38  | &     | 0x46 | 70  | F                 |
| 0x27 | 39  | 1     | 0x47 | 71  | G                 |
| 0x28 | 40  | (     | 0x48 | 72  | н                 |
| 0x29 | 41  | )     | 0x49 | 73  | I                 |
| 0x2A | 42  | *     | 0x4A | 74  | J                 |
| 0x2B | 43  | +     | 0x4B | 75  | K                 |
| 0x2C | 44  | 7     | 0x4C | 76  | L                 |
| 0x2D | 45  | -     | 0x4D | 77  | М                 |
| 0x2E | 46  |       | 0x4E | 78  | N                 |
| 0x2F | 47  | /     | 0x4F | 79  | 0                 |
| 0x30 | 48  | 0     | 0x50 | 80  | P                 |
| 0x31 | 49  | 1     | 0x51 | 81  | Q                 |
| 0x32 | 50  | 2     | 0x52 | 82  | R                 |
| 0x33 | 51  | 3     | 0x53 | 83  | S                 |
| 0x34 | 52  | 4     | 0x54 | 84  | т                 |
| 0x35 | 53  | 5     | 0x55 | 85  | U                 |
| 0x36 | 54  | 6     | 0x56 | 86  | 12 <mark>7</mark> |
| 0x37 | 55  | 7     | 0x57 | 87  | W                 |



| 1  | <pre>#include <xc.h> //PIC18F452</xc.h></pre>                  |
|----|--|
| 2  |  |
| 3  | // This program sends numeric values (hex/dec/bin)             |
| 4  | // for the ASCII characters of 0-5, A-D to PORTB               |
| 5  |  |
| 6  | void main(void)  |
| 7  | ₽ {  |
| 8  | <pre>unsigned char myNum[] = "012345ABCD"; //data stored</pre> |
| 9  | unsigned char z;   |
| 0  |  |
| 1  | TRISB = 0; //PORTB is an OUTPUT on all pins                    |
| 2  |  |
| 13 | for(z=0; z<10; z++)  |
| 4  |  |
| 15 | PORTB = myNum[z]; //write ASCII char to PORTB                  |
| 6  | }  |
| -  |  |

z = 6 PORTB = 'A' (in code) PORTB = 0x41 = 65 (actual)

**PORTB = 0b 0100 0001 (pins)** 

| Hex  | Dec | Char  | Hex  | Dec | Char              |
|------|-----|-------|------|-----|-------------------|
| 0x20 | 32  | Space | 0x40 | 64  | 6                 |
| 0x21 | 33  | 1     | 0x41 | 65  | A                 |
| 0x22 | 34  |       | 0x42 | 66  | В                 |
| 0x23 | 35  | #     | 0x43 | 67  | С                 |
| 0x24 | 36  | \$    | 0x44 | 68  | D                 |
| 0x25 | 37  | 8     | 0x45 | 69  | Е                 |
| 0x26 | 38  | &     | 0x46 | 70  | F                 |
| 0x27 | 39  | 1     | 0x47 | 71  | G                 |
| 0x28 | 40  | (     | 0x48 | 72  | н                 |
| 0x29 | 41  | )     | 0x49 | 73  | I                 |
| 0x2A | 42  | *     | 0x4A | 74  | J                 |
| 0x2B | 43  | +     | 0x4B | 75  | K                 |
| 0x2C | 44  | 7     | 0x4C | 76  | L                 |
| 0x2D | 45  | -     | 0x4D | 77  | М                 |
| 0x2E | 46  |       | 0x4E | 78  | N                 |
| 0x2F | 47  | 1     | 0x4F | 79  | 0                 |
| 0x30 | 48  | 0     | 0x50 | 80  | P                 |
| 0x31 | 49  | 1     | 0x51 | 81  | Q                 |
| 0x32 | 50  | 2     | 0x52 | 82  | R                 |
| 0x33 | 51  | 3     | 0x53 | 83  | S                 |
| 0x34 | 52  | 4     | 0x54 | 84  | т                 |
| 0x35 | 53  | 5     | 0x55 | 85  | U                 |
| 0x36 | 54  | 6     | 0x56 | 86  | 13 <mark>7</mark> |
| 0x37 | 55  | 7     | 0x57 | 87  | W                 |



### Signed char (-128 to +127)

• Still 8-bit data type but MSB is sign value

```
Write a C18 program to send values of -4 to +4 to Port B.
```



# Unsigned int (0 to 65,535)

- PIC18 is 8-bit architecture, **int type** (16 bits) takes two bytes of RAM (only use when necessary)
- C compilers use signed int (-32,768 to +32,767) by default unless we put "unsigned"
  - int == signed int

```
#include <P18F458.h>
void main(void)
{
    unsigned int z;
    TRISB = 0; //make Port B an output
    for(z=0;z<=50000;z++)
    {
        PORTB = 0x55;
        PORTB = 0xAA;
     }
    while(1); //stay here forever
}</pre>
```



### Larger Integer Types (short, long, short long)

Write a C18 program to toggle all bits of Port B 100,000 times.

Solution:

```
//toggle PB 100,00 times
#include <P18F458.h>
void main(void)
    unsigned short long z;
    unsigned int x;
    TRISE = 0;
                              //make Port B an output
    for(z=0;z=100000;z++)
        PORTB = 0x55:
        PORTB = 0xAA;
    while(1);
                              //stay here forever
```



- Can store and calculate numbers with decimals (precision)
- Always signed, can't be unsigned
  2.5, 32.05898, -1.00232, .2600313, 51156.01, etc.

#### TABLE 5-3: FLOATING-POINT DATA TYPES

| Туре        | Size (bits)    | Arithmetic Type |
|-------------|----------------|-----------------|
| float       | 24 or 32       | Real            |
| double      | 24 or 32       | Real            |
| long double | same as double | Real            |

• Further info: <u>Text</u> and <u>Video Explanation</u> <sup>17</sup>





- In C can use % to perform a modulus of two numbers (find the <u>whole number</u> remainder from a "repeated subtraction")
- 25 % 5 = 0
- 25 % 7 = 4
- 25 % 10 = 5
- 428 % 100 = 28
- 1568 % 10 = 8



```
int i = 7;
int j = 2;
int k = 0;
float f;
//through variables
k = i / j; // k =
f = i / j; // f = ?
f = (float)i / j; // f =
//direct numbers/literals
k = 7 / 2; // k =
f = 7 / 2; // f =
f = 7.0 / 2; // f =
```



```
int i = 7;
int j = 2;
int k = 0;
float f;
//through variables
k = i / j; // k = 3
f = i / j; // f = 3.0
f = (float)i / j; // f = 3.5
//direct numbers/literals
k = 7 / 2; // k = 3
f = 7 / 2; // f = 3.0
f = 7.0 / 2; // f = 3.5
```





• Want to have exact time differences or spacing between certain instructions

- Three methods:
  - Using a simple loop (for/while) (crude)
  - Using PIC18 timer peripheral (later)
  - Built-in delay functions (reliable and accurate)



### Two Factors for Delay Accuracy in C

### 1. The crystal's frequency (int. or ext.)

- Duration of clock period for instruction cycle

### 2. The compiler used for the C program

- In ASM, we control the exact instructions
- Different compilers produce different ASM code



### **Time Delay Example**

Write a C18 program to toggle all the bits of Port B ports continuously with a 250 ms delay. Assume that the system is PIC18F458 with XTAL = 10 MHz.

```
#include <PIC18F452.h>
void MS_Delay(unsigned int);
```

```
void main(void)
```

```
TRISB = 0;
while(1)
```

```
PORTB = 0x55;
MS_Delay(250);
PORTB = 0xAA;
MS_Delay(250);
```

void MS\_Delay(unsigned int msTime)

```
unsigned int i;
unsigned int j;
```

 $F_{OSC} = 10 \text{ MHz} = 10,000,000 \text{ cycles/sec}$ 

Each instruction takes 4 clock cycles (ticks)

 $F_{CY} = \text{Instruction Cycle Frequency} \\ = \frac{10MHZ}{4} = 2.5\text{MHz} = 2,500,000 \text{ Ins/sec}$ 

 $T_{CY} = \text{Instruction Cycle Time}$ = 1 / 2.5MHz = 0.0000004 sec per Ins = 0.0004 ms = 0.4 µs

How many IC (instructions) fit into 1ms? 1ms / 0.0004ms = 2,500

→ 2,500 Instruction Cycles take place in 1ms
 → 2,500 Instructions can complete in 1ms<sup>23</sup>



# Instruction Cycle



How many IC (instructions) fit into 1ms? 1 ms / 0.0004 ms = 2,500

- $\rightarrow$  2,500 Instruction Cycles take place in 1ms
- 24 2,500 Instructions can complete in 1ms (generalizing since most instructions only take 1 Ins. Cycle)  $\rightarrow$



# Delay Functions in the XC8 Compiler

- 1. Include the "xc.h" header file
- 2. Define your crystal's frequency
  - \_XTAL\_FREQ
- 3. Can now use these 2 delay functions:
  - \_\_\_\_\_delay\_us(x); //unsigned long (0 4294967295)
  - \_\_\_\_\_delay\_\_ms(x); //unsigned long (0 4294967295)

```
#include <xc.h> 
    #define XTAL FREQ 1000000
                                      // Running at 10MHz
    #define LED LEFT
                        PORTAbits.RA3 // QwikFlash red LED (left) to toggle
    #define LED CENTER PORTAbits.RA2 // QwikFlash red LED (center) to toggle
                        PORTAbits.RA1 // QwikFlash red LED (right) to toggle
    #define LED RIGHT
    void Toggle LEDs(void);
10
11
    void main (void)
12
       TRISA = 0; //PORTA is an OUTPUT
14
15
        //Main routine
       while(1)
16
17
18
            //Your main code goes here
19
            Toggle LEDs();
20
21
22
23
    void Toggle LEDs(void)
24
   🗏 🖣
       LED LEFT ^{=} 1;
25
          delay ms(100); 🔶 🛁
26
27
28
        LED CENTER ^{=} 1;
29
          delay ms(100);
31
        LED RIGHT ^{=} 1;
32
          delay ms(100);
33
```



- Btye-Size Register Access
  - Labels still the same
  - PORTA PORTD
  - TRISA TRISD
  - INTCON
- Bit-Addressable Register Access
  - PORTBbits.RB3
  - TRISCbits.RC7 or TRISCbits.TRISC7
  - INTCONbits.RBIE



# **PORT I/O Programming in C**



```
unsigned char mybyte;
TRISB = 0xFF; //Port B as input
TRISC = 0; //Port C as output
while(1)
{
    mybyte = PORTB; //get a byte from Port B
    MSDelay(500);
    PORTC = mybyte; //send it to Port C
}
```



### **PORTxbits.Rxy**

| Table 7-2: Single-Bit Addresses of PIC18F458/4580 Ports |       |       |       |       |            |  |
|---|-------|-------|-------|-------|------------|--|
| PORTA   | PORTB | PORTC | PORTD | PORTE | Port's Bit |  |
| RA0   | RB0   | RC0   | RD0   | RE0   | D0         |  |
| RA1   | RB1   | RC1   | RD1   | RE1   | D1         |  |
| RA2   | RB2   | RC2   | RD2   | RE2   | D2         |  |
| RA3   | RB3   | RC3   | RD3   |       | D3         |  |
| RA4   | RB4   | RC4   | RD4   |       | D4         |  |
| RA5   | RB5   | RC5   | RD5   |       | D5         |  |
|   | RB6   | RC6   | RD6   |       | D6         |  |
|   | RB7   | RC7   | RD7   |       | D7         |  |



```
#include <P18F458.h>
void MSDelay(unsigned int);
#define Dsensor PORTBbits.RB1
#define buzzer PORTCbits.RC7
void main(void)
                                   //PORTB.1 as an input
    TRISBbits.TRISB1 = 1;
                                   //make PORTC.7 an output
    TRISCbits.TRISC7 = 0;
    while (Dsensor == 1)
        buzzer = 0;
        MSDelay(200);
        buzzer = 1;
        MSDelay(200);
                             //stay here forever
    while(1);
```



Write a C18 program to get the status of bit RB0, and send it to RC7 continuously. Solution:

```
#include <P18F458.h>
#define inbit PORTBbits.RB0
#define outbit PORTCbits.RC7
void main(void)
  {
    TRISBbits.TRISB0 = 1; //make RB0 an input
    TRISCbits.TRISC7 = 0; //make RC7 an output
    while(1)
        {
        outbit = inbit; //get a bit from RB0
        //and send it to RC7
    }
}
```



### **.ASM Generated from C**

CSE@UTA

| 1:     |      | <pre>#include <p18f458.h></p18f458.h></pre> |                      |
|--------|------|---|----------------------|
| 2:     |      | #define inbit PORTBbits.RB0                 |                      |
| 3;     |      | #define outbit PORTCbits.RC7                |                      |
| 4:     |      | void main(void)                             |                      |
| 5:     |      | {   |                      |
| 6:     |      | TRISBbits.TRISB0 = 1;                       | //make RB0 an input  |
| 0000E2 | 8093 | BSF 0xf93, 0, ACCESS                        |                      |
| 7:     |      | TRISCUITS.TRISC7 = $0;$                     | //make RC7 an output |
| 0000E4 | 9E94 | BCF 0xf94, 0x7, ACCESS                      |                      |
| 8:     |      | while(1)                                    |                      |
| 0000F2 | D7F9 | BRA 0xe6                                    |                      |
| 9:     |      | {   |                      |
| 10:    |      | outbit = inbit;                             | //get bit from RBO   |
| 0000E6 | 5081 | MOVF 0xf81, W, ACCESS                       |                      |
| 0000E8 | 0B01 | ANDLW 0x1                                   |                      |
| 0000EA | E002 | BZ 0xf0                                     |                      |
| 0000EC | 8E82 | BSF 0xf82, 0x7, ACCESS                      |                      |
| 0000EE | D001 | BRA 0xf2                                    |                      |
| 0000F0 | 9E82 | BCF 0xf82, 0x7, ACCESS                      |                      |
| 11:    |      |   | //and send it to RC7 |
| 12:    |      | }   |                      |
| 13:    |      | }   |                      |
| 0000F4 | 0012 | RETURN 0                                    | 3                    |



### **Header Files**

- Remember that certain register/variable names are <u>not native C keywords</u>
- They are PIC-specific
   PORTB, TRISA, TMR0H, PRODL, etc.
- Defined and mapped in header file
   Using regular data types (char, int, struct, etc.)
- Regular P18Fxxx.h (device) header files
   C:\Program Files (x86)\Microchip\xc8\v1.20\include



### **Header Files**

- Other functional headers are available
  - adc.h
  - delays.h
  - i2c.h
  - pwm.h
  - timers.h
  - usart.h



- Peripheral library Header Files
  - C:\Program Files (x86)\Microchip\xc8\v1.20\include\plib
  - C:\Program Files (x86)\Microchip\xc8\v1.20\sources\pic18\plib



# **Logic Operations in C**

### • Bit-Wise Operators

| Table 7-3: Bit-wise Logic Operators for C |   |     |    |       |          |  |
|---|---|-----|----|-------|----------|--|
|   |   | AND | OR | EX-OR | Inverter |  |
| Α   | В | A&B | AB | A^B   | Y=~B     |  |
| 0   | 0 | 0   | 0  | 0     | 1        |  |
| 0   | 1 | 0   | 1  | 1     | 0        |  |
| 1   | 0 | 0   | 1  | 1     |          |  |
| 1   | 1 | 1   | 1  | 0     |          |  |

- Bit-Wise Shift Operators
  - Can shift right/left by X bits
     Shift right >>
     Shift left <<</li>



# **Logic Operations in C**

| TRISB | =   | 0;           | //make Ports B, C,                 |
|-------|-----|--------------|------------------------------------|
| TRISC | =   | 0;           | //and D output ports               |
| TRISD | ÷   | 0;           |                                    |
| PORTB | =   | 0x35 & 0x0F; | //ANDing                           |
| PORTC | =   | 0x04   0x68; | //ORing                            |
| PORTD | =   | 0x54 ^ 0x78; | //XORing                           |
| PORTB | =   | ~0x55;       | //inverting                        |
| PORTC | =   | 0x9A >> 3;   | //shifting right 3 times           |
| PORTD | -   | 0x77 >> 4;   | //shifting right 4 times           |
| PORTB | =   | 0x6 << 4;    | <pre>//shifting left 4 times</pre> |
| while | (1) | i            | //stay here forever                |
|       |     |              |                                    |





### Binary (hex) to Decimal and ASCII Conversion

- Sometimes we can't handle multiple-digit decimals natively in C for display purposes
- printf() is standard for generic C but requires more memory space than a PIC18 is willing to sacrifice
- Best to build your own "custom" print or display functions in C



• Want each digit of **253** (0b11111101, 0xFD) and convert to ASCII for displaying



 Want each digit of 253 (0b11111101, 0xFD) and convert to ASCII for displaying Hex Dec Char Hex I 0x20 32 Space 0x40

| 1 | unsigned char whole, part, d1, d2, d3; |
|---|--|
| 2 |  |
| 3 | whole = 253; //whole == d3_d2_d1       |
| Δ |  |

| 0x20       32       Space       0x40       64       @         0x21       33       !       0x41       65       A         0x22       34       "       0x42       66       B         0x23       35       #       0x43       67       C         0x24       36       \$       0x44       68       D         0x25       37       %       0x45       69       E         0x26       38       &       0x46       70       F         0x27       39       '       0x47       71       G         0x28       40       (       0x48       72       H         0x29       41       )       0x49       73       I         0x20       42       *       0x40       74       J         0x22       41       )       0x42       76       L         0x20       45       -       0x40       77       M         0x22       46       .       0x42       78       N         0x22       47       /       0x44       74       J         0x31       49       1       0x50       8  | Hex  | Dec | Char  | Hex         | Dec | Char |
|--|------|-----|-------|-------------|-----|------|
| 0x21 $33$ ! $0x41$ $65$ A $0x22$ $34$ " $0x42$ $66$ B $0x23$ $35$ # $0x43$ $67$ C $0x24$ $36$ \$ $0x44$ $68$ D $0x25$ $37$ % $0x45$ $69$ E $0x26$ $38$ $&$ $0x46$ $70$ F $0x27$ $39$ ' $0x47$ $71$ G $0x27$ $39$ ' $0x47$ $71$ G $0x28$ $40$ ( $0x48$ $72$ H $0x29$ $41$ ) $0x49$ $73$ I $0x28$ $42$ * $0x44$ $74$ J $0x28$ $43$ + $0x48$ $75$ K $0x22$ $44$ , $0x44$ $78$ N $0x22$ $45$ - $0x54$ $80$ P $0x30$ $48$ 0 $0x55$ $85$ U $0x31$ $49$ 1 $0x51$ $81$ Q $0x33$ $51$ 3 $0x55$ $85$ U $0x36$ $54$ 6 $0x56$ $86$ V $0x33$ $55$ 7 $0x57$ $87$ W $0x38$ $56$ 8 $0x58$ $88$ X   | 0x20 | 32  | Space | 0x40        | 64  | 6    |
| 0x22 $34$ " $0x42$ $66$ B $0x23$ $35$ # $0x43$ $67$ C $0x24$ $36$ \$ $0x44$ $68$ D $0x25$ $37$ \$ $0x44$ $69$ E $0x26$ $38$ \$ $0x45$ $69$ E $0x27$ $39$ ' $0x47$ $71$ G $0x27$ $39$ ' $0x47$ $71$ G $0x27$ $39$ ' $0x47$ $71$ G $0x28$ $40$ ( $0x48$ $72$ H $0x29$ $41$ ) $0x49$ $73$ I $0x24$ $42$ * $0x44$ $74$ J $0x28$ $43$ + $0x48$ $75$ K $0x22$ $44$ , $0x44$ $76$ L $0x22$ $45$ - $0x44$ $77$ M $0x22$ $44$ , $0x44$ $78$ N $0x22$ $44$ , $0x44$ $78$ N $0x24$ $47$ / $0x44$ $78$ N $0x25$ $47$ / $0x44$ $78$ N $0x31$ $49$ 1 $0x51$ $81$ Q $0x33$ $51$ 3 $0x53$ $83$ S $0x34$ $52$ 4 $0x56$ $86$ V $0x36$ $54$ 6 $0x56$ $86$ V $0x33$ $55$ 7 $0x57$ $87$ W $0x38$ $56$ 8 $0x58$ $88$ X </td <td>0x21</td> <td>33</td> <td>1</td> <td>0x41</td> <td>65</td> <td>А</td>  | 0x21 | 33  | 1     | 0x41        | 65  | А    |
| 0x23 $35$ # $0x43$ $67$ C $0x24$ $36$ \$ $0x44$ $68$ D $0x25$ $37$ \$ $0x45$ $69$ E $0x26$ $38$ \$ $0x45$ $69$ E $0x27$ $39$ ' $0x47$ $71$ G $0x27$ $39$ ' $0x47$ $71$ G $0x28$ $40$ ( $0x48$ $72$ H $0x29$ $41$ ) $0x49$ $73$ I $0x24$ $42$ * $0x44$ $74$ J $0x28$ $43$ + $0x48$ $75$ K $0x22$ $44$ , $0x44$ $76$ L $0x22$ $45$ - $0x40$ $77$ M $0x22$ $46$ . $0x44$ $79$ O $0x30$ $48$ 0 $0x50$ $80$ P $0x31$ $49$ 1 $0x51$ $81$ Q $0x33$ $51$ 3 $0x53$ $83$ S $0x34$ $52$ 4 $0x54$ $84$ T $0x35$ $53$ 5 $0x55$ $85$ U $0x36$ $54$ 6 $0x56$ $86$ V $0x38$ $56$ 8 $0x58$ $88$ X $0x39$ $57$ 9 $0x59$ $89$ Y $0x3A$ $58$ $12$ $0x54$ $30$ $2$  | 0x22 | 34  |       | 0x42        | 66  | в    |
| 0x24       36       \$       0x44       68       D         0x25       37       %       0x45       69       E         0x26       38       &       0x46       70       F         0x27       39       '       0x47       71       G         0x28       40       (       0x48       72       H         0x29       41       )       0x49       73       I         0x2A       42       *       0x4A       74       J         0x2B       43       +       0x4B       75       K         0x2C       44       ,       0x4C       76       L         0x2D       45       -       0x4D       77       M         0x2E       46       .       0x4F       79       O         0x30       48       0       0x50       80       P         0x31       49       1       0x51       81       Q         0x33       51       3       0x53       83       S         0x34       52       4       0x54       84       T         0x35       53       5       0x55       85 <td>0x23</td> <td>35</td> <td>#</td> <td>0x43</td> <td>67</td> <td>С</td>       | 0x23 | 35  | #     | 0x43        | 67  | С    |
| 0x25       37       %       0x45       69       E         0x26       38       &       0x46       70       F         0x27       39       '       0x47       71       G         0x28       40       (       0x47       71       G         0x29       41       )       0x49       73       I         0x28       42       *       0x44       74       J         0x28       42       *       0x48       75       K         0x22       41       )       0x48       75       K         0x22       43       +       0x48       76       L         0x20       45       -       0x40       77       M         0x22       46       .       0x44       79       O         0x30       48       0       0x50       80       P         0x31       49       1       0x51       81       Q         0x32       50       2       0x52       82       R         0x33       51       3       0x53       83       S         0x34       52       4       0x54       84 <td>0x24</td> <td>36</td> <td>\$</td> <td>0x44</td> <td>68</td> <td>D</td>       | 0x24 | 36  | \$    | 0x44        | 68  | D    |
| 0x26 $38$ $&$ $0x46$ $70$ $F$ $0x27$ $39$ ' $0x47$ $71$ $G$ $0x28$ $40$ ( $0x48$ $72$ $H$ $0x29$ $41$ ) $0x49$ $73$ $I$ $0x2A$ $42$ * $0x4A$ $74$ $J$ $0x2B$ $43$ + $0x4B$ $75$ $K$ $0x2C$ $44$ , $0x4C$ $76$ $L$ $0x2D$ $45$ - $0x4D$ $77$ $M$ $0x2E$ $46$ . $0x4E$ $78$ $N$ $0x2F$ $47$ / $0x4F$ $79$ $O$ $0x30$ $48$ $0$ $0x50$ $80$ $P$ $0x31$ $49$ 1 $0x51$ $81$ $Q$ $0x33$ $51$ $3$ $0x53$ $83$ $S$ $0x34$ $52$ $4$ $0x54$ $84$ $T$ $0x35$ $53$ $5$ $0x55$ $85$ $U$ $0x36$ $54$ $6$ $0x56$ $86$ $V$ $0x37$ $55$ $7$ $0x57$ $87$ $W$ $0x38$ $56$ $8$ $0x58$ $88$ $X$ $0x39$ $57$ $9$ $0x54$ $39$ $Y$  | 0x25 | 37  | 8     | 0x45        | 69  | Е    |
| 0x27 $39$ ' $0x47$ $71$ $G$ $0x28$ $40$ ( $0x48$ $72$ $H$ $0x29$ $41$ ) $0x49$ $73$ $I$ $0x2A$ $42$ * $0x48$ $74$ $J$ $0x2B$ $43$ + $0x4B$ $75$ $K$ $0x2C$ $44$ , $0x4C$ $76$ $L$ $0x2D$ $45$ - $0x4D$ $77$ $M$ $0x2E$ $46$ . $0x4E$ $78$ $N$ $0x2F$ $47$ / $0x4F$ $79$ $O$ $0x30$ $48$ $0$ $0x50$ $80$ $P$ $0x31$ $49$ $1$ $0x51$ $81$ $Q$ $0x33$ $51$ $3$ $0x53$ $83$ $S$ $0x33$ $51$ $3$ $0x53$ $83$ $S$ $0x34$ $52$ $4$ $0x54$ $84$ $T$ $0x35$ $53$ $5$ $0x55$ $85$ $U$ $0x36$ $54$ $6$ $0x56$ $86$ $V$ $0x37$ $55$ $7$ $0x57$ $87$ $W$ $0x38$ $56$ $8$ $0x58$ $88$ $X$ $0x39$ $57$ $9$ $0x53$ $90$ $Z$  | 0x26 | 38  | &     | 0x46        | 70  | F    |
| 0x28       40       (       0x48       72       H         0x29       41       )       0x49       73       I         0x2A       42       *       0x48       74       J         0x2B       43       +       0x4B       75       K         0x2C       44       ,       0x4C       76       L         0x2D       45       -       0x4D       77       M         0x2E       46       .       0x4E       78       N         0x2F       47       /       0x4F       79       O         0x30       48       0       0x50       80       P         0x31       49       1       0x51       81       Q         0x33       51       3       0x53       83       S         0x34       52       4       0x54       84       T         0x35       53       5       0x55       85       U         0x36       54       6       0x56       86       V         0x37       55       7       0x57       87       W         0x38       56       8       0x58       88 <td>0x27</td> <td>39</td> <td>1</td> <td>0x47</td> <td>71</td> <td>G</td>        | 0x27 | 39  | 1     | 0x47        | 71  | G    |
| 0x29       41       )       0x49       73       I         0x2A       42       *       0x4A       74       J         0x2B       43       +       0x4B       75       K         0x2C       44       ,       0x4D       76       L         0x2D       45       -       0x4D       77       M         0x2E       46       .       0x4E       78       N         0x2F       47       /       0x4F       79       O         0x30       48       0       0x50       80       P         0x31       49       1       0x51       81       Q         0x32       50       2       0x52       82       R         0x33       51       3       0x53       83       S         0x34       52       4       0x54       84       T         0x35       53       5       0x55       85       U         0x36       54       6       0x56       86       V         0x37       55       7       0x57       87       W         0x38       56       8       0x58       88 <td>0x28</td> <td>40</td> <td>(</td> <td><b>0x48</b></td> <td>72</td> <td>н</td> | 0x28 | 40  | (     | <b>0x48</b> | 72  | н    |
| 0x2A       42       *       0x4A       74       J         0x2B       43       +       0x4B       75       K         0x2C       44       ,       0x4C       76       L         0x2D       45       -       0x4D       77       M         0x2E       46       .       0x4F       79       O         0x2F       47       /       0x4F       79       O         0x30       48       0       0x50       80       P         0x31       49       1       0x51       81       Q         0x33       51       3       0x53       83       S         0x33       51       3       0x55       85       U         0x33       51       3       0x55       85       U         0x35       53       5       0x55       85       U         0x36       54       6       0x56       86       V         0x37       55       7       0x57       87       W         0x38       56       8       0x58       88       X         0x39       57       9       0x54       39 <td>0x29</td> <td>41</td> <td>)</td> <td>0x49</td> <td>73</td> <td>I</td>        | 0x29 | 41  | )     | 0x49        | 73  | I    |
| 0x2B       43       +       0x4B       75       K         0x2C       44       ,       0x4C       76       L         0x2D       45       -       0x4D       77       M         0x2E       46       .       0x4E       78       N         0x2F       47       /       0x4F       79       O         0x30       48       0       0x50       80       P         0x31       49       1       0x51       81       Q         0x32       50       2       0x52       82       R         0x33       51       3       0x53       83       S         0x34       52       4       0x54       84       T         0x35       53       5       0x55       85       U         0x36       54       6       0x56       86       V         0x37       55       7       0x57       87       W         0x38       56       8       0x58       88       X         0x39       57       9       0x59       89       Y  | 0x2A | 42  | *     | 0x4A        | 74  | J    |
| 0x2C       44       ,       0x4C       76       L         0x2D       45       -       0x4D       77       M         0x2E       46       .       0x4E       78       N         0x2F       47       /       0x4F       79       O         0x30       48       0       0x50       80       P         0x31       49       1       0x51       81       Q         0x32       50       2       0x52       82       R         0x33       51       3       0x53       83       S         0x34       52       4       0x54       84       T         0x35       53       5       0x55       85       U         0x36       54       6       0x56       86       V         0x37       55       7       0x57       87       W         0x38       56       8       0x58       88       X         0x39       57       9       0x54       39       Y  | 0x2B | 43  | +     | 0x4B        | 75  | K    |
| 0x2D       45       -       0x4D       77       M         0x2E       46       .       0x4E       78       N         0x2F       47       /       0x4F       79       O         0x30       48       0       0x50       80       P         0x31       49       1       0x51       81       Q         0x32       50       2       0x52       82       R         0x33       51       3       0x53       83       S         0x34       52       4       0x54       84       T         0x35       53       5       0x55       85       U         0x36       54       6       0x56       86       V         0x37       55       7       0x57       87       W         0x38       56       8       0x58       88       X         0x39       57       9       0x57       89       Y         0x3A       58       :       0x5A       20       Z  | 0x2C | 44  | 7     | 0x4C        | 76  | L    |
| 0x2E       46       .       0x4E       78       N         0x2F       47       /       0x4F       79       O         0x30       48       0       0x50       80       P         0x31       49       1       0x51       81       Q         0x32       50       2       0x52       82       R         0x33       51       3       0x53       83       S         0x34       52       4       0x54       84       T         0x35       53       5       0x55       85       U         0x36       54       6       0x56       86       V         0x37       55       7       0x57       87       W         0x38       56       8       0x58       88       X         0x39       57       9       0x58       89       Y         0x3A       58       :       0x5A       20       Z  | 0x2D | 45  | -     | 0x4D        | 77  | М    |
| 0x2F       47       /       0x4F       79       0         0x30       48       0       0x50       80       P         0x31       49       1       0x51       81       Q         0x32       50       2       0x52       82       R         0x33       51       3       0x53       83       S         0x34       52       4       0x54       84       T         0x35       53       5       0x55       85       U         0x36       54       6       0x56       86       V         0x37       55       7       0x57       87       W         0x38       56       8       0x58       88       X         0x39       57       9       0x58       89       Y         0x3A       58       :       0x5A       20       Z  | 0x2E | 46  |       | 0x4E        | 78  | N    |
| 0x30       48       0       0x50       80       P         0x31       49       1       0x51       81       Q         0x32       50       2       0x52       82       R         0x33       51       3       0x53       83       S         0x34       52       4       0x54       84       T         0x35       53       5       0x55       85       U         0x36       54       6       0x56       86       V         0x37       55       7       0x57       87       W         0x38       56       8       0x58       88       X         0x39       57       9       0x58       89       Y         0x3A       58       :       0x5A       20       Z  | 0x2F | 47  | /     | 0x4F        | 79  | 0    |
| 0x31       49       1       0x51       81       Q         0x32       50       2       0x52       82       R         0x33       51       3       0x53       83       S         0x34       52       4       0x54       84       T         0x35       53       5       0x55       85       U         0x36       54       6       0x56       86       V         0x37       55       7       0x57       87       W         0x38       56       8       0x58       88       X         0x39       57       9       0x59       89       Y         0x3A       58       :       0x5A       200       Z   | 0x30 | 48  | 0     | 0x50        | 80  | Р    |
| 0x32       50       2       0x52       82       R         0x33       51       3       0x53       83       S         0x34       52       4       0x54       84       T         0x35       53       5       0x55       85       U         0x36       54       6       0x56       86       V         0x37       55       7       0x57       87       W         0x38       56       8       0x58       88       X         0x39       57       9       0x59       89       Y         0x3A       58       :       0x5A       200       Z   | 0x31 | 49  | 1     | 0x51        | 81  | Q    |
| 0x33       51       3       0x53       83       S         0x34       52       4       0x54       84       T         0x35       53       5       0x55       85       U         0x36       54       6       0x56       86       V         0x37       55       7       0x57       87       W         0x38       56       8       0x58       88       X         0x39       57       9       0x59       89       Y         0x3A       58       :       0x5A       200       Z   | 0x32 | 50  | 2     | 0x52        | 82  | R    |
| 0x34       52       4       0x54       84       T         0x35       53       5       0x55       85       U         0x36       54       6       0x56       86       V         0x37       55       7       0x57       87       W         0x38       56       8       0x58       88       X         0x39       57       9       0x59       89       Y         0x3A       58       :       0x5A       20       Z  | 0x33 | 51  | 3     | 0x53        | 83  | S    |
| 0x35       53       5       0x55       85       U         0x36       54       6       0x56       86       V         0x37       55       7       0x57       87       W         0x38       56       8       0x58       88       X         0x39       57       9       0x59       89       Y         0x3A       58       :       0x5A3@0       Z  | 0x34 | 52  | 4     | 0x54        | 84  | т    |
| 0x36       54       6       0x56       86       V         0x37       55       7       0x57       87       W         0x38       56       8       0x58       88       X         0x39       57       9       0x59       89       Y         0x3A       58       :       0x5A       200       Z   | 0x35 | 53  | 5     | 0x55        | 85  | U    |
| 0x37       55       7       0x57       87       W         0x38       56       8       0x58       88       X         0x39       57       9       0x59       89       Y         0x3A       58       :       0x5A       300       Z   | 0x36 | 54  | 6     | 0x56        | 86  | v    |
| 0x38         56         8         0x58         88         X           0x39         57         9         0x59         89         Y           0x3A         58         :         0x5A         300         Z   | 0x37 | 55  | 7     | 0x57        | 87  | W    |
| 0x39 57 9 0x59 89 Y<br>0x3A 58 : 0x5A 3 0 Z  | 0x38 | 56  | 8     | 0x58        | 88  | Х    |
| 0x3A 58 : 0x5A300 Z  | 0x39 | 57  | 9     | 0x59        | 89  | Y    |
|  | 0x3A | 58  | :     | 0x5A        | 390 | Z    |



 Want each digit of 253 (0b11111101, 0xFD) and convert to ASCII for displaying Hex Dec Char Hex D 0x20 32 Space 0x40

| 1 | unsigned char whole | , part, | d1, d2 | 2, d3;   |
|---|---------------------|---------|--------|----------|
| 2 |                     |         |        |          |
| 3 | whole = 253; //whol | e == d3 | d2 d1  |          |
| 4 |                     | _       |        |          |
| 5 | part = whole / 10;  | //part  | = 253  | / 10 = 2 |
| 6 | d1 = whole % 10;    | //d1    | = 253  | 8 10 = 3 |
| 7 | d2 = part % 10;     | //d2    | = 25   | 8 10 = 5 |
| 8 | d3 = part / 10;     | //d3    | = 25   | / 10 = 2 |

| Hex     | Dec | Char  | Hex         | Dec | Char         |
|---------|-----|-------|-------------|-----|--------------|
| 0x20    | 32  | Space | 0x40        | 64  | 6            |
| 0x21    | 33  | 1     | 0x41        | 65  | А            |
| 0x22    | 34  |       | 0x42        | 66  | в            |
| 0x23    | 35  | #     | 0x43        | 67  | С            |
| 0x24    | 36  | \$    | 0x44        | 68  | D            |
| 0x25    | 37  | 8     | 0x45        | 69  | Е            |
| 0x26    | 38  | &     | 0x46        | 70  | F            |
| 0x27    | 39  | 1.00  | 0x47        | 71  | G            |
| 0x28    | 40  | (     | <b>0x48</b> | 72  | н            |
| 0x29    | 41  | )     | 0x49        | 73  | I            |
| 0x2A    | 42  | *     | 0x4A        | 74  | J            |
| 0x2B    | 43  | +     | 0x4B        | 75  | K            |
| 0x2C    | 44  | ,     | 0x4C        | 76  | L            |
| 0x2D    | 45  | -     | 0x4D        | 77  | М            |
| 0x2E    | 46  |       | 0x4E        | 78  | N            |
| 0x2F    | 47  | 1     | 0x4F        | 79  | 0            |
| 0x30    | 48  | 0     | 0x50        | 80  | Р            |
| 0x31    | 49  | 1     | 0x51        | 81  | Q            |
| 0x32    | 50  | 2     | 0x52        | 82  | R            |
| 0x33    | 51  | 3     | 0x53        | 83  | S            |
| 0x34    | 52  | 4     | 0x54        | 84  | т            |
| 0x35    | 53  | 5     | 0x55        | 85  | U            |
| 0x36    | 54  | 6     | 0x56        | 86  | v            |
| 0x37    | 55  | 7     | 0x57        | 87  | W            |
| 0x38    | 56  | 8     | 0x58        | 88  | Х            |
| 0x39    | 57  | 9     | 0x59        | 89  | Y            |
| 0x3A    | 58  | :     | 0x5A        | 4غ  | $\mathbf{Z}$ |
| · · · · |     |       |             |     | -            |



 Want each digit of 253 (0b11111101, 0xFD) and convert to ASCII for displaying Hex Dec Char Hex D 0x20 32 Space 0x40

| 1  | unsigned char whole            | , part, d1, d2, d3;   |    |
|----|--------------------------------|-----------------------|----|
| 2  |                                |                       |    |
| 3  | <pre>whole = 253; //whol</pre> | $e == d3_{d2}d1$      |    |
| 4  |                                |                       |    |
| 5  | <pre>part = whole / 10;</pre>  | //part = 253 / 10 = . | 25 |
| 6  | d1 = whole % 10;               | //d1 = 253 % 10 = .   | 3  |
| 7  | d2 = part % 10;                | //d2 = 25 % 10 = .    | 5  |
| 8  | d3 = part / 10;                | //d3 = 25 / 10 = .    | 2  |
| 9  |                                |                       |    |
| 10 | d1 = d1 + 48;                  | //or + 0x30           |    |
| 11 | d2 = d2 + 48;                  | //or + 0x30           |    |
| 12 | d3 = d3 + 48;                  | //or + 0x30           |    |

| Hex  | Dec | Char  | Hex  | Dec | Char |
|------|-----|-------|------|-----|------|
| 0x20 | 32  | Space | 0x40 | 64  | 6    |
| 0x21 | 33  | 1     | 0x41 | 65  | А    |
| 0x22 | 34  |       | 0x42 | 66  | в    |
| 0x23 | 35  | #     | 0x43 | 67  | С    |
| 0x24 | 36  | \$    | 0x44 | 68  | D    |
| 0x25 | 37  | 8     | 0x45 | 69  | E    |
| 0x26 | 38  | &     | 0x46 | 70  | F    |
| 0x27 | 39  | 1     | 0x47 | 71  | G    |
| 0x28 | 40  | (     | 0x48 | 72  | н    |
| 0x29 | 41  | )     | 0x49 | 73  | I    |
| 0x2A | 42  | *     | 0x4A | 74  | J    |
| 0x2B | 43  | +     | 0x4B | 75  | K    |
| 0x2C | 44  | 7     | 0x4C | 76  | L    |
| 0x2D | 45  | -     | 0x4D | 77  | М    |
| 0x2E | 46  |       | 0x4E | 78  | N    |
| 0x2F | 47  | /     | 0x4F | 79  | 0    |
| 0x30 | 48  | 0     | 0x50 | 80  | Р    |
| 0x31 | 49  | 1     | 0x51 | 81  | Q    |
| 0x32 | 50  | 2     | 0x52 | 82  | R    |
| 0x33 | 51  | 3     | 0x53 | 83  | S    |
| 0x34 | 52  | 4     | 0x54 | 84  | т    |
| 0x35 | 53  | 5     | 0x55 | 85  | U    |
| 0x36 | 54  | 6     | 0x56 | 86  | v    |
| 0x37 | 55  | 7     | 0x57 | 87  | W    |
| 0x38 | 56  | 8     | 0x58 | 88  | х    |
| 0x39 | 57  | 9     | 0x59 | 89  | Y    |
| 0x3A | 58  |       | 0x5A | 490 | Z    |



### **#define Directive**

Can associate labels with numbers or registers as a constant

### #define LED\_OUTPUT PORTBbits.RB2 #define MAX\_USERS 50





- For PIC C Programming
   Textbook Ch. 7 for more details
- Start looking over Arithmetic/Logic
  - Textbook Ch. 5