# EC-310 Microprocessor and Microcontroller Based Design

## Chapter - 7

# Delay Programming in C

**Nazar Abbas Saqib**
nazar.abbas@ceme.nust.edu.pk

# Outline

1. Data Types and Time Delays in C.
2. I/O Programming in C.
3. Logic Operations in C.
4. Data Conversion Programs in C.
5. Data Serialization in C.
6. Program ROM Allocation in C18.
7. Data RAM Allocation in C18.

# Why Programming in C?

- Programming in assembly is tedious whereas programming in C is very easy.
- Programming in C consumes less time as compared to programming in assembly.
- C code is easier to modify and update as compared with assembly.
- Well established libraries of C.
- C code is portable to other microcontrollers with very less modifications.

# Data Types and Time Delays in C

- **Hex file** generated for **C code** is **larger** as compared to hex file generated for assembly code.
- The size of the hex file generated for same code may differ depending upon the compiler being used.
- To optimize the size constraint for C code, proper use of data types is very important.
- A good understanding of C data types for PIC18 can help programmers to create a smaller hex files.

# Data Types and Time Delays in C

**Table 7-1: Some Data Types Widely Used by C18**

| Data Type | Size in Bits | Data Range/Usage |
|---|---|---|
| unsigned char | 8-bit | 0 to 255 |
| char | 8-bit | –128 to +127 |
| unsigned int | 16-bit | 0 to 65,535 |
| int | 16-bit | –32,768 to +32,767 |
| unsigned short | 16-bit | 0 to 65,535 |
| short | 16-bit | –32,768 to +32,767 |
| unsigned short long | 24-bit | 0 to 16,777,215 |
| short long | 24-bit | –8,388,608 to +8,388,607 |
| unsigned long | 32-bit | 0 to 4,294,967,295 |
| long | 32-bit | –2,147,483,648 to +2,147,483,648 |

# Data Types and Time Delays in C

- **Unsigned Char**
- The character data is **most natural data type** for 8-bit microcontrollers.
- The unsigned char is an **8-bit** data type.
- It can take values from **00H to FFH** i.e (0 – 255).
- It is one of the most widely used data types for PIC18.
- C compilers use **signed char as default**, unless we put the keyword unsigned in front of char.
- Unsigned char can also be used for string of ASCII characters.

# Data Types and Time Delays in C

- **Unsigned Char**

- In declaring variables, pay careful attention to the size of data and try to use unsigned char instead of int if possible.

- Since PIC18 microcontroller has a limited number of registers and data RAM locations, using int in place of char can lead to a larger size hex file.

# Data Types and Time Delays in C

**Example 7-1**

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
  }
```

Run the above program on your simulator to see how Port B displays values 00–FFH in binary. Notice that "while(1)" is needed if this program is running in hardware.

# Data Types and Time Delays in C

**Example 7-2**

Write a C18 program to send hex values for ASCII characters of 0, 1, 2, 3, 4, 5, A, B, C, and D to Port B.

**Solution:**

```
#include <P18F458.h>
void main(void)
  {
    unsigned char mynum[]= "012345ABCD";//data is stored in RAM
    unsigned char z;
    TRISB = 0;                    //make Port B an output
    for(z=0;z<10;z++)
      PORTB = mynum[z];
    while(1);                     //stay here forever
  }
```

Run the above program on your simulator to see how Port B displays values 30H, 31H, 32H, 33H, 34H, 35H, 41H, 42H, 43H, and 44H (the hex values for ASCII 0, 1, 2, etc.). Notice that the last statement "while(1)" is needed only if we run the program in hardware. This is like "GOTO $" or "BRA $" in Assembly language.

# Data Types and Time Delays in C

**Example 7-3**

Write a C18 program to toggle all the bits of Port B continuously.

**Solution:**

```
// Toggle PB forever
#include <P18F458.h>
void main(void)
  {
    TRISB = 0;              //make Port B an output
    for(;;)                 //repeat forever
      {
        PORTB = 0x55;       //0x indicates the data is in hex (binary)
        PORTB = 0xAA;
      }
}
```

Run the above program on your simulator to see how Port B toggles continuously.

# Data Types and Time Delays in C

- **Signed Char**
- The signed char is an **8-bit** data type.
- MSB is used to represent – or + value.
- For –ve numbers, MSB is 1.
- For +ve numbers, MSB is 0.
- 7-bits are only used for magnitude of the signed numbers.
- Range: -128 to +127
- By default, char is of signed nature.

# Data Types and Time Delays in C

**Example 7-4**

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

**Solution:**

```
//sign numbers
#include <P18F458.h>
void main(void)
  {
    char mynum[]= {+1,-1,+2,-2,+3,-3,+4,-4};
    unsigned char z;
    TRISB = 0;                    //make Port B an output
    for(z=0;z<8;z++)
      PORTB = mynum[z];
    while(1);                     //stay here forever
  }
```

Run the above program on your simulator to see how PORTB displays values of 1, FFH, 2, FEH, 3, FDH, 4, and FCH (the hex values for +1, –1, +2, –2, etc.). See Chapter 5 for discussion of signed numbers.

# Data Types and Time Delays in C

- **Unsigned int**
- The unsigned int is a **16-bit** data type.
- Range: 0000H – FFFFH i.e (0 to 65535)
- In PIC18, unsigned int is used to define 16-bit variables such as:
  - Memory addresses
  - Counters for values greater than 256.
- Use of int data type must be avoided since PIC18 is an 8-bit microcontroller and int is a 16-bit data type. This would result in a greater size hex file.
- Situations where there is no need of signed data, we should use unsigned int instead of signed int. This provides us with much wider range.

# Data Types and Time Delays in C

- **Signed int**
- The signed int is a **16-bit** data type.
- MSB is the sign bit.
- For negative numbers, MSB is 1.
- For positive numbers, MSB is 0.
- For magnitude we have 15-bits.
- Range: -32768 to 32767.

# Data Types and Time Delays in C

- **Other data types in PIC18**
- C18 compiler supports both **short long** and **long** data types for values **greater than 16-bit**.
- Short long value is 24-bits wide.
- Long is 32-bits wide.

# Data Types and Time Delays in C

**Example 7-5**

Write a C18 program to toggle all bits of Port B 50,000 times.

**Solution:**

```
#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
  }
```

Run the above program on your simulator to see how Port B toggles continuously. Notice that the maximum value for unsigned int is 65,535.

# Data Types and Time Delays in C

**Example 7-6**

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;
    TRISB = 0;                    //make Port B an output
    for(z=0;z<=100000;z++)
      {
        PORTB = 0x55;
        PORTB = 0xAA;
      }
    while(1);                     //stay here forever
  }
```

# Time Delays in C

- There are two ways to create a time delay in C18:
  - Using simple for loop.
  - Using PIC8 timers.

**Delays must be verified using oscilloscopes.**

- At this stage, we'll discuss delays, using for loop. Delays using timers are discussed in chapter 9.

- Two factors can affect the accuracy of the delay:
  - The **crystal frequency** to OSC1 and OSC2 input pins is the most important factor in time delay calculation.
  - The second factor is the **compiler used to compile the C program**. In assembly we can control the exact instructions and their sequences used in delays. In C, the compiler converts the C statements to assembly instructions. Different compilers may convert the same C code in two different assembly codes.

**Example 7-7**

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.

**Solution:**

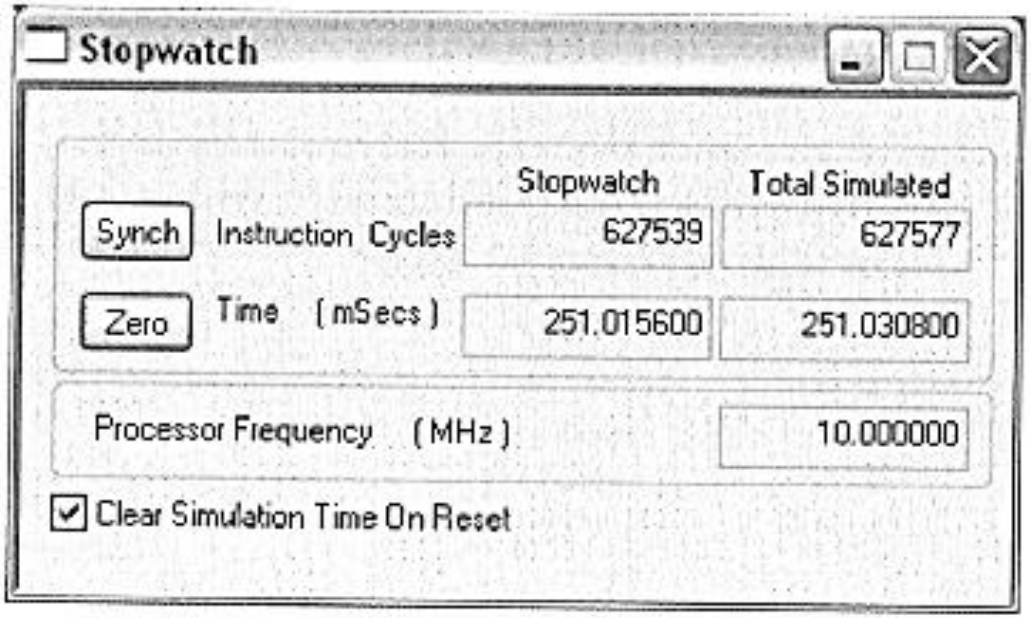
```
#include <P18F458.h>
void MSDelay(unsigned int);
void main(void)
  {
    TRISB = 0;                    //make Port B an output
    while(1)                      //repeat forever
      {
        PORTB = 0x55;
        MSDelay(250);
        PORTB = 0xAA;
        MSDelay(250);
      }
  }

void MSDelay(unsigned int itime)
  {
    unsigned int i; unsigned char j;
    for(i=0;i<itime;i++)
      for(j=0;j<165;j++);
  }
```

**Example 7-8**

Write a C18 program to toggle all the bits of Port C and Port D continuously with a 250 ms delay.

**Solution:**

```
//this program is tested for the PIC18F458 with XTAL = 10 MHz
#include <P18F458.h>
void MSDelay(unsigned int);
void main(void)
  {
    TRISC = 0;
    TRISD = 0;                    //make Ports C and D output
    while(1)                      //another way to do it forever
      {
        PORTC = 0x55;
        PORTD = 0x55;
        MSDelay(250);
        PORTC = 0xAA;
        PORTD = 0xAA;
        MSDelay(250);
      }
  }
void MSDelay(unsigned int itime)
  {
    unsigned int i; unsigned char j;
    for(i=0;i<itime;i++)
      for(j=0;j<165;j++);
  }
```




MPLAB's simulator has a stop-watch function that allows us to view the time delay before we program the microcontroller.

**Figure 7-2. Time Delay Measurement for Example 7-8 Using MPLAB**