

Introduction to C Programming with Embedded Systems Part II



ME430: MECHATRONICS

C Programming Language

- Created in 1972
- Many languages built on C
 - C++, Java, C#, Objective C



ME430: MECHATRONICS

Declaring variables required in C

- Declaring variables

```
// Just declaring variables
char x;
unsigned char y;
int Dave, Steve, a, b, c, d, e, f, g, h;
long Bob;
float Steve;
char arrayBob[10];
```

- Initializing variable with declaration

```
// Initialized data
int x = 0;
float f = 1.253;
char message1[] = "Handy string syntax"
int numlist[] = {1, 1, 2, 3, 5, 8, 13, 21, 34};
```

We'll talk about all the types and details later



ME430: MECHATRONICS

Variable scope

- Global vs. Local Variables

```

/** Global Variables *****
int sampleVariable;

*****
* Function:      void main(void)
*****
#pragma code
void main (void)
{
    int x; // Local variable x - scope limited to within main

    while (1)
    {
    }
}

*****
* Additional Helper Functions
*****

*****
* Function:      void sample(void)
* Input Variables: none
* Output Return: none
* Overview:      Use a comment block like this before functions
*****
void sampleFunction()
{
    char Bob; // Local variable Bob - scope limited to within sampleFunction
}

```



ME430: MECHATRONICS

Variable scope

Global Variables

- Defined outside of any function (within Global variables section)
- Variable can be used anywhere within the .c file
- Variable does not need to be passed to functions
- Excellent for interrupts (explain interrupts later in course)

Local Variables

- Defined within functions (always FIRST thing in function!)
 - Causes a syntax error if you try declaring variable mid function
- Need to be passed by functions
- Helps organize your code
- Better programming practice BUT the debugger sometimes doesn't handle well



ME430: MECHATRONICS

Control structures

- Concepts you know, syntax is new
- **if** conditional statement
 - **switch** conditional statement (may be new)
- **for** loops
- **while** loops



ME430: MECHATRONICS

if – Standard if else structure

```
// if - else statement
if (Bob == 12)
{
    Dave = 5;
    PORTB = 0x03;
}
else
{
    Dave = 3;
    PORTB = 0x01;
}
```



ME430: MECHATRONICS

if – Beware common errors

<pre>// Basic if statement if (Bob == 10) Dave = 5; // BAD if statment if (Bob = 10) Dave = 5; // BAD if statement if(Bob == 10); Dave = 5;</pre>	<pre>// Multiple lines if statment if (Bob == 11) { Dave = 5; PORTB = 0x03; } // BAD multiple lines if statement if (Bob == 11) Dave = 5; PORTB = 0x03;</pre>
---	--

Let's practice an if statement using the simulator!



ME430: MECHATRONICS

if – Your turn

- Make a variable **age**
 - char age = 10;
- Make an if-else statement that prints
 - “age is greater than 15\n” or
 - “age is less than or equal to 15\n”
 - as appropriate
- Then change the variable age to 20
 - Rebuild and run the program again



ME430: MECHATRONICS

if – Multiple Conditions

```
// Multiple conditions
if ( (Bob == 11) && (x < 3) )
{
    Dave = 5;
    PORTB = 0x03;
}

// BAD statement
if ( 2 < x < 6 )
{
    Dave = 5;
    PORTB = 0x03;
}

// Conditional Statements == != < <= > >=
// Logical Operators      && ||
```



ME430: MECHATRONICS

if – Using else if

```
// if - else if - else if - else statement
if (Bob == 12)
{
    Dave = 5;
    PORTB = 0x03;
}
else if (Bob == 13)
{
    Dave = 4;
    PORTB = 0x02;
}
else
{
    Dave = 3;
    PORTB = 0x01;
}
```



ME430: MECHATRONICS

switch

```
// Switch statement
switch (Bob)
{
    case 3:
        Dave = 3;
        break;
    case 4:
        Dave = 4;
        break;
    case 5:
        Dave = 50;
        break;
    default:
        Dave = 0;
        break;
}
```

Convenience function with weird syntax. Handy for discrete value conditions

Can ALWAYS be done with an [else if](#) series of statements



ME430: MECHATRONICS

for

```

int i;

for ( i=0 ; i<10 ; i=i+1)
{
    printf("i = %d\n", i);
}

```

```

for ( initialization ; conditional statement ; update variable statement )
{
}

```



ME430: MECHATRONICS

for – Standard shortcut

```

int i;

for ( i=0 ; i<10 ; i=i+1)
{
    printf("i = %d\n", i);
}

```

Basic for loop

```

for ( i=0 ; i<10 ; i++)
{
    printf("i = %d\n", i);
}

```

Same loop using shortcut operator

Only useful for *for* loops that increment the variable by 1. Very common though.



ME430: MECHATRONICS

for – Your turn

- Write a **for** loop
 - runs from **num** = 20 to num = 0
 - decrements num by 5 each time
 - i.e. num should be 20, 15, 10, 5, and 0
- Inside the for loop, put an “**if**”-“**else if**”-“**else**” that prints (as appropriate)
 - num is greater than 15
 - num is equal to 15
 - num is less than 15



while

```
// A while loop that runs forever
while (1)
{
    // This area loops forever
}

// A while loop implementing a for loop
Bob = 20;
while ( Bob < 35)
{
    printf("Bob is %d years old", Bob);
    Bob++;
}
```



Functions

```

/** Local Function Prototypes *****/
void sampleFunction(void);

/*****
 * Additional Helper Functions
 *****/

/*****
 * Function:      void sample(void)
 * Input Variables: none
 * Output Return: none
 * Overview:      Use a comment block like this before functions
 *****/
void sampleFunction()
{
    // Some function that does a specific task
}

```



ME430: MECHATRONICS

C programming references

- Only scratching the surface of entire language
- More in depth references:
 - Essential C: An introduction
 - <http://cslibrary.stanford.edu/101/EssentialC.pdf>
 - Programming in C (4th Edition) – Steve Kochan
 - Thousands more on the web
 - (plenty of time since 1972 for posts!)

Use the rest of the time to work on Labs



ME430: MECHATRONICS