

2nd C Class with Mr. Czik

Counting Systems

0	1	2	3	4	5	6	7	8	9	10	Arabic
	I	II	III	IV	V	VI	VII	VIII	IX	X	Roman

Numerals – Glyphs

Base 10 – Decimal – we have 10 fingers

1000	100	10	1	1 1000, 2 100s, 3 10s, and 4 1s
1	2	3	4	1000+200+30+ 4 = 1234 (decimal number)

Other number bases can be useful – how about 12 – 12 months

- 12 in a Dozen – 12 x 12 in a Gross – 12 Hours twice in a day
- 12 inches in a foot
- 12 (2, 3, 4, 6) factors better than 10 (2 & 5 only)

Complex numbering systems can be understood even by young children

Time 12:00 or 24:00 clocks, 60 minutes increments hours, etc

Computers work in Binary (Base 2) – Zero and One (true and false)

128	64	32	16	8	4	2	1
1	1	1	1	1	1	1	1

Binary number **11111111**, 128+64+32+16+8+4+2+1 = 255 (decimal numbers)

Programmers use Hexadecimal (Base 16, Hex = 6, Decimal = 10) numbers to help represent binary numbers without using as many digits

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	Hex
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Dec

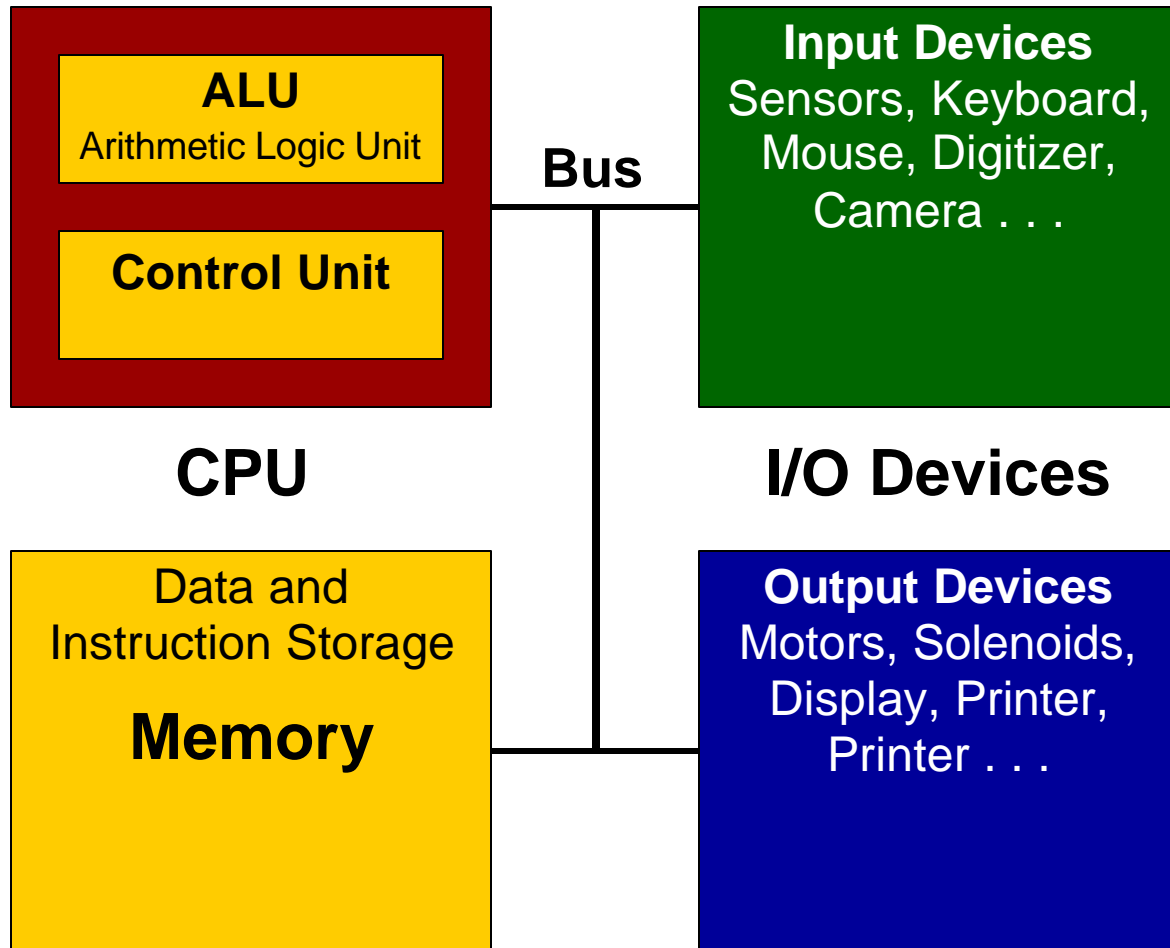
4096	256	16	1	1 4096, 2 256s, 3 16s, and 4 1s
1	2	3	4	4096+512+48+ 4 = 4660 (decimal number)

2nd C Class with Mr. Czik

Binary Chart – with Decimal and Hexadecimal values

128	64	32	16	8	4	2	1		Dec	Hex
							0		0	0
							1		1	1
						1	0		2	2
						1	1		3	3
					1	0	0		4	4
					1	0	1		5	5
					1	1	0		6	6
					1	1	1		7	7
				1	0	0	0		8	8
				1	0	0	1		9	9
				1	0	1	0		10	A
				1	0	1	1		11	B
				1	1	0	0		12	C
				1	1	0	1		13	D
				1	1	1	0		14	E
				1	1	1	1		15	F
			1	0	0	0	0		16	10
			1	0	0	0	1		17	11
			1	0	0	1	0		18	12
			1	0	0	1	1		19	13
			1	0	1	0	0		20	14
			1	0	1	0	1		21	15
			1	0	1	1	0		22	16
			1	0	1	1	1		23	17
			1	1	0	0	0		24	18
			1	1	0	0	1		25	19
			1	1	0	1	0		26	1A
			1	1	0	1	1		27	1B
			1	1	1	0	0		28	1C
			1	1	1	0	1		29	1D
			1	1	1	1	0		30	1E
			1	1	1	1	1		31	1F
		1	0	0	0	0	0		32	20

128	64	32	16	8	4	2	1		Dec	Hex
-----	----	----	----	---	---	---	---	--	-----	-----



Parts of the Computer

CPU

Computes in binary numbers, which can represent program instructions, numbers or alphanumeric characters

Runs programs (Control Unit) and calculates math (ALU)

A standard CPU will run only one instruction at a time

Memory

Used to store data and program instructions, either on a temporary working basis (RAM) or long-term (disk drives, flash memory)

I/O (Input/Output)

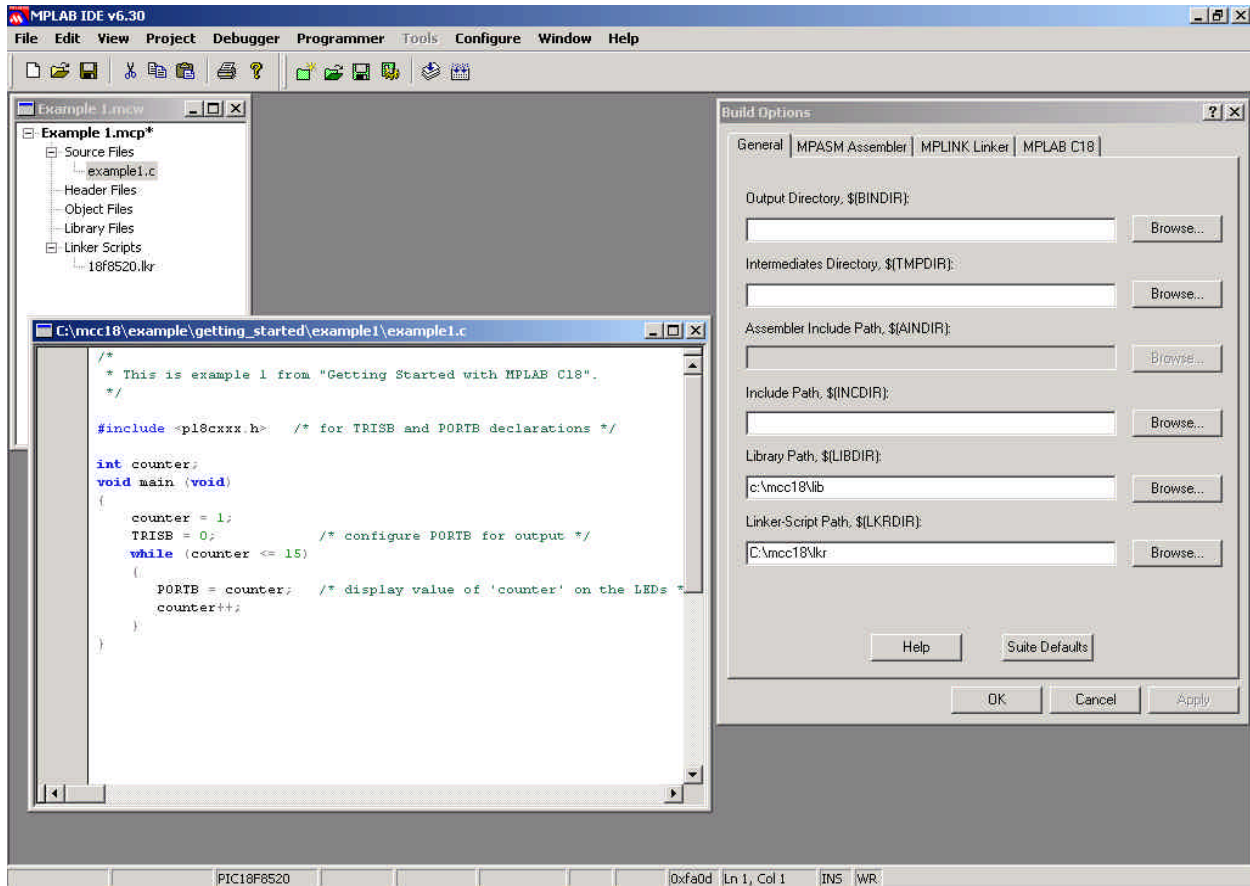
Devices that communicate between the computer and the real world

Bus

Interconnections between the different parts of the computer

2nd C Class with Mr. Czik

Using the MPLAB IDE (Interactive Development Environment) to Program in C



First, configure the MPLAB IDE with the Project Wizard

Project Wizard - Next

Device – select **PIC18F8520** – Next

Active Toolsuite – select **Microchip C18 Toolsuite**

Location – enter **C:\mcc18\bin\cpp18.exe** if needed – Next

Project Name – enter **Example 1**

Project Directory – Browse to

C:\mcc18\example\getting_started\example1 - Next

Skip adding files – Next

At Summary, press Finish

2nd C Class with Mr. Czik

Set Project Build Options

Project – Build Options – Project

Library path should be set to **c:\mcc18\lib**

Linker path should be set to **c:\mcc18\lkr**

OK

Add 18f8520.lkr to Project

Right click on **Linker Scripts**

Choose to Add Files

Browse to c:\mcc18\lkr

Select 18f8520.lkr, and Open

Once configured, add example 1.c file to Project

Right click on **Source Files**

Choose to **Add Files**

Select **example 1.c**, and Open

Double click on example 1.c to open the file for examination and editing

Note that the text between /* and */ is just for commenting your code, and does not run at any time. The MPLAB IDE shows these comments in green, and declarations and statements in bold blue, with the rest of the text in unbolded black.

C is a compiled language, which means that it is turned into a machine language format before it is run (binary code is loaded into the Robovation controller, not the C code in text format).

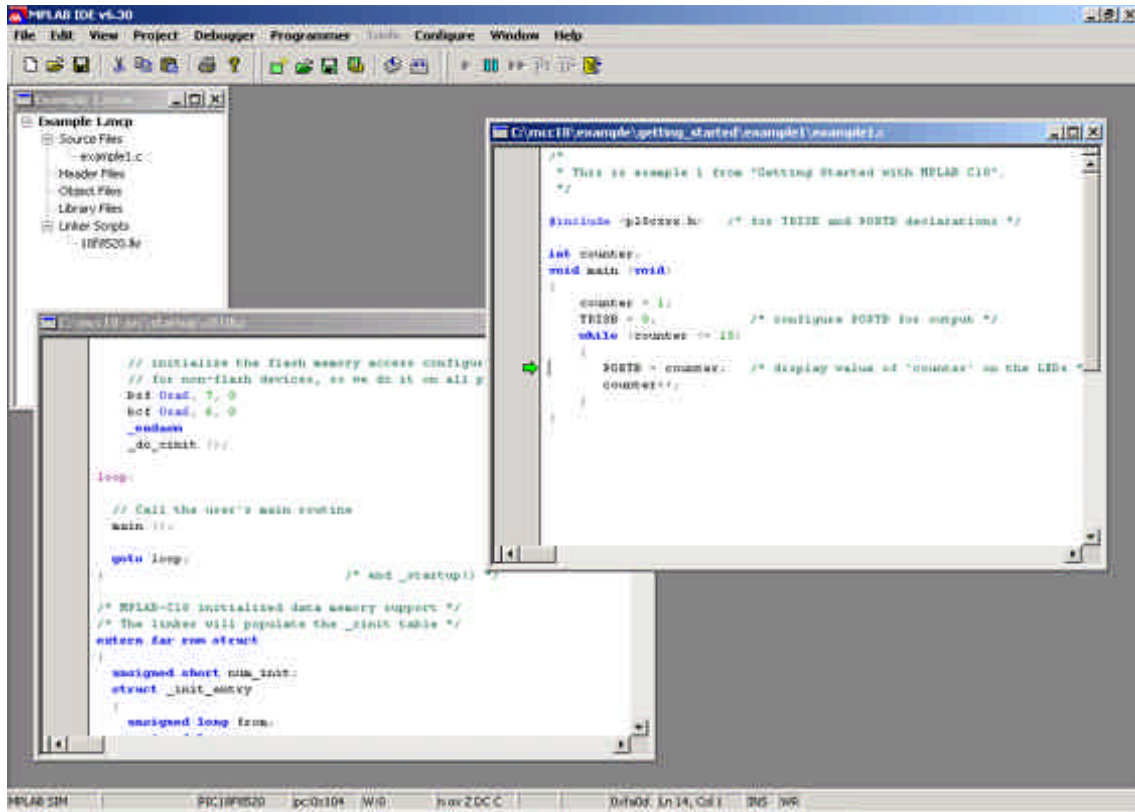
RIS 2.0 for LEGO Mindstorms is an interpreted language, in that the Mindscripts code that is produced from the graphical programming environment is loaded onto the RCX and converted into machine code as each line is run. Compilers run much faster than interpreters.

An example of Mindscripts code (which is somewhat C-like):

2nd C Class with Mr. Czik

```
program test {
  #include <RCX2.h>
  #include <RCX2MLT.h>
  #include <RCX2Sounds.h>
  #include <RCX2Def.h>
  sensor touch1 on 1
  touch1 is switch as boolean
  main {
    ext InterfaceType "kFreestyle"
    rcx_ClearTimers
    bbs_GlobalReset([A B C])
    try {
      if counter1 > 10{
        while touch1 is closed {
          on [ A B C ] for 100
        }
      }
      else
      {
        if counter1 > 20{
          repeat 3 {
          }
        }
        else
        {
        }
      }
    } retry on fail
  }
}
```


2nd C Class with Mr. Czik



```
Example Lamp
Source Files
  - example1.c
Header Files
Object Files
Library Files
Library Scripts
  - I185200.kl

C:\mclab\example\getting_started\example1\example1.c
/*
 * This is example 1 from "Getting Started with MPLAB C18".
 */
#include <stdio.h> /* for printf and fprintf declarations */

int counter;
void main (void)
{
    counter = 1;
    printf("counter = %d\n", counter); /* configure printf for output */
    while (counter <= 15)
    {
        printf("counter = %d\n", counter); /* display value of 'counter' on the LED */
        counter++;
    }
}

// initialize the flash memory access config
// for non-flash devices, so we do it on all y
#define Dead 7, 0
#define Dead 6, 0
#ifdef
do_init();

loop:

// Call the user's main routine
main();

goto loop; /* and _startup() */

/* MPLAB-C18 initialized data memory support */
/* The linker will populate the _init table */
extern far rom struct
{
    unsigned short nix_init;
    struct _init_memory
    unsigned long from;
}

MPLAB21M | PIC18F520 | pc0x10x | W0 | hex20CC | Default In3, Col1 | 06 | 06
```

The Goto statement can be used to unconditionally jump from one location in the program code to another. While it can be a quick and dirty solution to program problems, Goto statements should be avoided as the enemy of structured, understandable code. Goto statements encourage “spaghetti” code, difficult to decipher. Use indents, comments, and proper program flow for structured programming.

Goto label

label

Controlling program flow is accomplished by using the following statements:

if if-else switch break continue goto