PROG-2000 What you need to know about programming the Z8

1. **Hardware**
    1.1. Configuration registers (PAADDR, PACTL, PCADDR, PCCTL)
        1.1.1. Subregisters for:
            1.1.1.1. Data Direction (0x01), 0 = Output, 1 = Input
            1.1.1.2. Alternate Function (0x02), 0 = Alt Fcn Disabled, 1 = Alt Fcn Enabled
            1.1.1.3. Output Control (0x03), 0 = Push-Pull (Normal), 1 = Open Drain
    1.2. Data Registers
        1.2.1. PAOUT connects to active-LOW LEDs (0 = ON, 1 = OFF)
        1.2.2. PCIN connects to a pushbutton switch (pressed = 0, not pressed = 1)
2. **Software**
    2.1. Integrated Development Environment (IDE)
        2.1.1. Making a Project File (*.zdsproj)
            2.1.1.1. Select correct kit (…KITG)
            2.1.1.2. Select correct programming interface (USB SmartCable)
        2.1.2. Making a C source file (*.c)
        2.1.3. Adding the C file to the Project (Project, Add files to Project)
        2.1.4. Compiling/building a program
        2.1.5. Reset the program
        2.1.6. Run the program (Go)
        2.1.7. Debugging tools
            2.1.7.1. Setting a breakpoint and running program to a breakpoint
            2.1.7.2. Single-stepping
            2.1.7.3. Observing variables in a watch window
            2.1.7.4. Observing PORT variables in a Special Function Registers window
    2.2. C Programming
        2.2.1. #include <ez8.h>
        2.2.2. int main(void) { }
        2.2.3. Declaring and initializing variables (unsigned char, int, etc.)
        2.2.4. Port configuration statements (See 1.1.1., above)
        2.2.5. Flow control statements (while, if, else, for)
        2.2.6. Assignment statements (leds = 0x02; PAOUT = leds; pushbutton = PCIN;)
        2.2.7. Comparison statements (if(x == 4) {do this;})
        2.2.8. return statements

Common typos in C:

The first place to look for errors in a C program is in common typographical errors. Some of the most common are:

- missing semicolon (;) at the end of a line
- missing /* */ or // for comments
- unbalanced () or {}
- case error (recall that myvar and MyVar are not the same)
- spelling error (e.g., **int main(vois)** instead of **int main(void)**)

Tip: If the compiler returns an error message for code on line *x*, try looking on an earlier line. For example, if an error is indicated on line 5, but line 5 looks correct, look for an error on line 4 or earlier. Sometimes the compiler will try to make sense of an error by continuing to read code on later lines. If this does not work, the program will crash, but maybe not exactly where the error occurred.