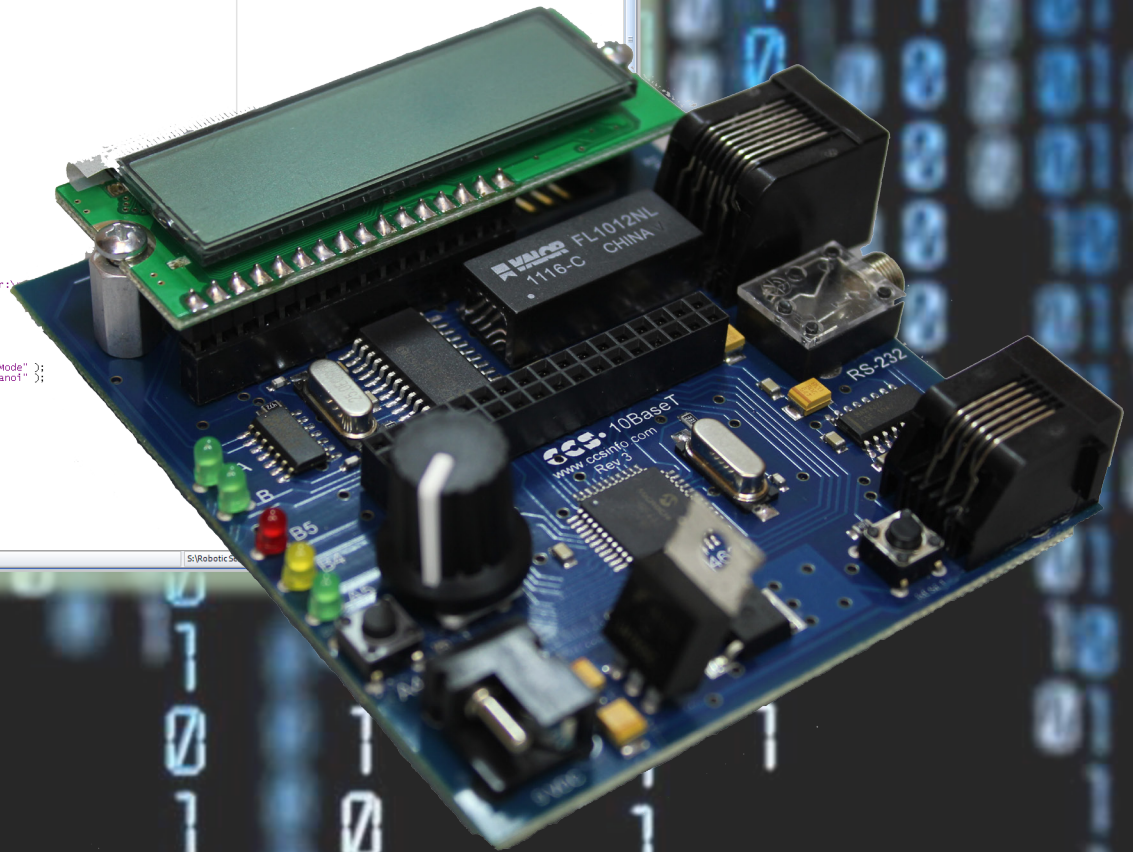
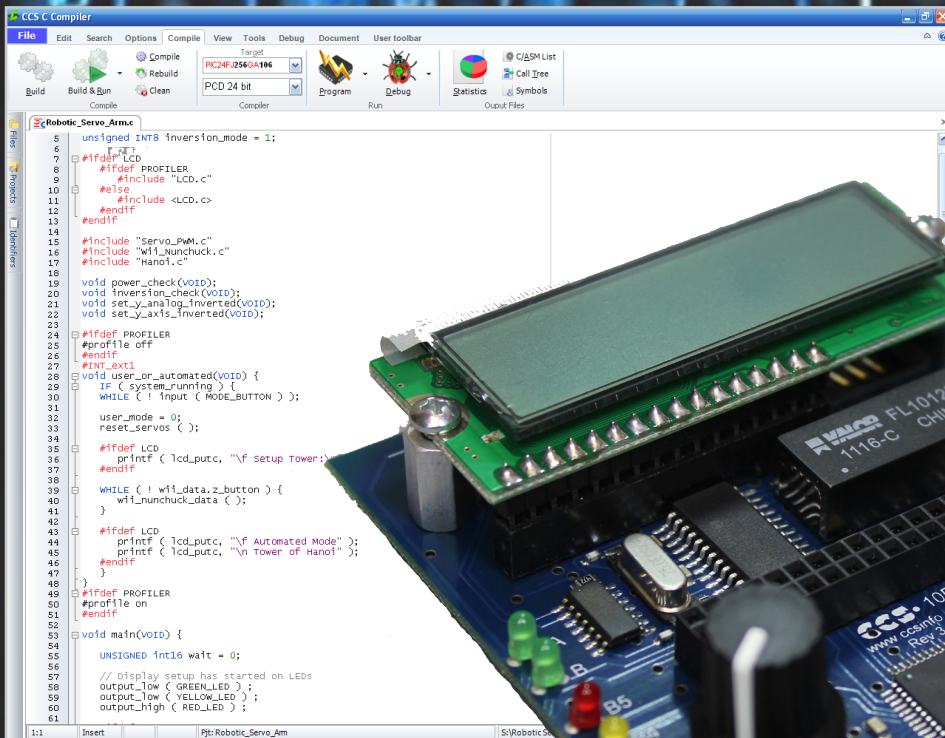


Professional - Powerful Microchip PIC[®] MCU Development Tool Solutions



PIC[®] MCU and dsPIC[®] are registered trademarks of Microchip Technology, Inc. in the U.S. and other countries.



www.ccsinfo.com

sales@ccsinfo.com

262-522-6500

sales x35

support x32

CCS C Version 5 - The C Code Conqueror

New features of V5 include:

- Numerous built-in libraries that generate relevant and tight code for a specific application. For example: RS232, RS485, PWM, timers, Capacitive and much more.
- IDE editor integrated with C to provide advanced editing such as finding a symbol definition, finding all references to an ID, bookmarking across files, background compiling with syntax marking, syntax helping, column editing and support for regular expressions.
- Multiple compiler versions can be installed and versions can be selected in the IDE. This allows an easy way to fix a project on a specific version for testing and certification.
- C Profiler Tool to track time and usage information on functions, code blocks and receive live data from running programs.
- CCS Data Streaming allows an ICD unit to be used so program I/O can be routed to a PC window inside and without the debugger.
- C++ style CIN/COU I/O streams with full data formatting to any device or for strings.
- Identifier explorer to easily show the usage and relationship of all program identifiers.
- Graphical viewing of program statistics and the call tree.
- Easy function to copy a project, zip up all project files for archive, or to quickly copy the hex file to a standard location.
- Various options for saving all source code changes. Easy viewing in the IDE to show changes between source code versions.
- Fast native Win32 IDE with no .net, and no Java, and fully integrated for Win8.

Processor & Peripheral Controls

The CCS C Compiler for PIC10, PIC12, PIC14, PIC16, PIC18 and PIC24 microcontrollers has over 307 Built-in Functions to access PIC® MCU hardware easily and producing efficient and highly optimized code.

Functions such as timers, A/D, EEPROM, SSP, PSP, USB, I2C and more:

- Built-in libraries that work with all chips for RS-232 serial I/O, I2C, discrete I/O and precision delays
- Serial I/O functions allow standard functions such as GETC() and PRINTF() to be used for RS-232 like I/O
- Formatted printf allows for easy formatting and display in HEX or decimal
- Multiple I2C and RS232 ports may be easily defined
- #use rs232() offers options to specify a maximum wait time for getc
- Hardware transceiver used when possible, but for all other occasions the compiler generates a software serial transceiver
- Microcontroller clock speed may be specified in a PRAGMA to permit built-in functions to delay for a given number of microseconds or milliseconds
- Functions such as INPUT() and OUTPUT_HIGH() properly maintain the tri-state registers
- Compiler directives determine if tri-state registers are refreshed on every I/O or if the I/O is as fast as possible
- #USE SPI ()
- Simple functions like READ_ADC() to read a value from A/D converter
- Source code drivers included for LCD modules, keypads, 24xx and 94xx serial EEPROM, X10, DS1302 and NJU6355 real time clocks, Dallas touch memory devices, DS2223 and PCF8570, LTC1298 and PCF8591 A/D converters, temperature sensors, digital pots, I/O expander and much more

Advanced Functions

The compiler can handle inline or separate functions, as well as parameter passing in re-usable registers. Transparent to the user, the compiler handles calls across pages automatically and analyzes program structure and call tree to optimize RAM and ROM usage.

Additional features include:

- Efficient function implementation allow call trees deeper than the hardware stack
- Automatic linking handles multiple code pages
- Assembly code may be inserted anywhere in the source and may reference C variables
- Function Overloading allows for several functions with the same name, but differences in number and type of parameters
- Default Parameters can be used in a function if arguments are not used in a call
- Interrupt functions supported on PCM/PCH. The compiler generates all start-up and clean-up code as well as identifying the correct function to be called
- Reference parameters may be used to improve code readability and inline function efficiency
- Generation of multiple HEX files for chips with external memory
- Variable Number Of Parameters in a function
- Relocatable Objects / Multiple Compilation Unit (IDE Only)
- Automatic #fuses Configuration

CCS C Compiler - Code Optimization at its core!

CCS C Compiler is comprised with Standard C operators and built-in libraries that are specific to PIC® MCU registers, and access to hardware features from C. Exclusive features include:

PIC10 / PIC12 / PIC14 / PIC16 / PIC18

1, 8, 16, 32-bit integer types & 32-bit floating point
 Bit Arrays and Fixed Point Decimals
 #BIT and #BYTE will allow C variables to be placed at absolute addresses to map registers to C variables
 Standard one-bit type (Short Int) permits the compiler to generate very efficient Bit-oriented code
 Constants (including strings and arrays) are saved in program memory
 Flexible Handling of Constant Data
 Variable length Constant Strings
 AddressMod capability to create user defined address spaces in memory device

Advanced Features in PIC24 & dsPIC® DSCs

Also 48 & 64-bit floating point for calculations requiring greater precision or broader range
 #BIT, #BYTE and #WORD will allow C variables to be placed at absolute addresses to map registers
 Constants packed in all bytes of ROM
 Enhanced oscillator control to choose from a multitude of clock sources, PLL and power saving options
 Function recursion allows for interactive processing algorithms

Example Programs

A large number of example programs are included with the software. The following is a list of many of the programs. Most programs will work with any chip by just changing the #INCLUDE line that includes the device information. All of the following programs have wiring instructions at the beginning of the code in a comment header. The SLOW.EXE program included in the program directory may be used to demonstrate the example programs and uses a PC COM port to communicate with the target.

LCD	Frequency counter	Fixed Point	DTMF Tones	Boot Loader	MOD Bus
A/D	7 Seg LED	TCP/IP	CRC Calculator	CAN Bus	LIN Bus
PWM	Data Logger	Floating Point	CCP	I/O for 8-Pin Parts	RFID
Comparator	Pattern Generator	ICD Debugging	Watchdog Timer	Sleep	SPI
PSP	Stepper Motors	Advanced Macros	Analog Comparator	Timers	RTOS
Interrupts	Tone Generation	Memory Management	Optical Encoder	Web Server	Serial
Magnetic Card Reader	Sensors	I ² C	USB	E-mailer	

Generic header files are included for the standard PIC® microcontrollers. These files are in the DEVICES directory. The pins of the chip are defined in these files in the form PIN_B2. It is recommended that for a given project, the file is copied to a project header file and the PIN_xx defines be changed to match the actual hardware. For example; LCDRW (matching the mnemonic on the schematic). Use the generic include files by placing the following in your main .C file:

Included C Drivers

SERIAL EEPROM/FLASH	STANDARD C	USB	SOUNDS	NETWORKING/ INTERNET	RFID	OTHER
2041	ERRNO	USBN960X	ISD4003	TCP	EM4095	Digital Compass
24xx	ASSERT	PIC_USB	TONES	PPP	EM4402	Keypad
25xx	FLOAT	PIC18_USB	WTS701	S7600	EM4150	Mag Card Reader
93xx	LIMITS	ANALOG	LCD MODULE	RTL8019	EXPANDED I/O	PLL Interface
AT2421	SETJMP	14KCAL	GLCD	ENC28J60	74165	Dallas One Wire
AT25256	STDDEF	68HC68R1	KS0108	REAL-TIME CLOCK	74595	IR Decoder
CE51X	STDIO	AD7705	LCD420	DS1302	MAX7300	Line Tracker
CE62X	STDLIB	AD7715	SED1335	NJU6355	SC28L19x	Servo Control
CE67X	DIGITAL POT	ADS8320	HDM64GS12	DS1305	SERIAL RAM	X10
9512	AD8400	LTC1298	RS232	ISL1209	68HC68R1	Cyclic Redundancy Code
MMC/SD	DS1868	MAX517	INPUT	CAN FUNCTIONALITY	68HC68R2	RS485
FLOATEE	MCP41010	MCP3208	LOADER	MCP251x	PCF8570	N9085UD
STRING FUNCTIONS	TEMPERATURE	TRIG	ACCELEROMETER	8xxx8	M68AF031	PNI11096
STDLIB	DS1621	MATH	ADXL210	18F4580	PCF8570	LMX2326
STRING	DS1621M				D41256	TOUCH
	DS1631				MT4264	Dallas Touch
	DS1624					
	LM75CIM3					

Built-in Functions that Maximize Code Efficiency

Built-In Functions

CODE PROFILER

profileout()

RS-232

fprintf()
getc()
getch()
gets()
kbhit()
perror()
printf()
putchar()
puts()
setup_uart()

SPI TWO-WIRE I/O

setup_spi1()
setup_spi2()
setup_spi3()
spi_data_is_in()
spi_data_is_in2()
spi_data_is_in3()
spi_init()
spi_prewrite()
spi_read_16()
spi_read()
spi_read2_16()
spi_read2()
spi_read3_16()
spi_read3()
spi_read4_16()
spi_speed()
spi_write_16()
spi_write()
spi_write2_16()
spi_write2()
spi_write3_16()
spi_write3()
spi_write4_16()
spi_xfer()

VOLTAGE REF

setup_comparator()*
setup_low_volt_detect()
setup_vref()

ANALOG

adc_done()
adc_done2()
dac_write()
read_adc()
read_adc2()
set_adc_channel()
set_adc_channel2()
setup_adc_ports()
setup_adc_ports2()
setup_adc()
setup_adc2()
setup_dac()
setup_opamp1()
setup_opamp2()
setup_port_a()

STANDARD C SPECIAL

bsearch()
mblen()
mbstowcs()
mbtowc()
qsort()
rand()
srand()
_va_arg()
va_arg()
va_end()
va_start()
wcstombs()
wctomb()

I²C

i2c_isr_state()
i2c_poll()
i2c_read()
i2c_slaveaddr()
i2c_speed()
i2c_start()
i2c_stop()
i2c_write()

STANDARD C MATH

abs()
atof()
atof48()
atof64()
atoi()
atoi32()
atoi48()
atoi64()
div()
ldiv()

STANDARD C MEMORY

calloc()
free()
get_motor_pwm_count()
longjmp()
malloc()
memchr()
memcmp()
memcpy()
memmove()
memset()
offsetof()
offsetofbit()
realloc()
set_motor_pwm_duty()
set_motor_pwm_event()
set_motor_unit()
setjmp()
setup_motor_pwm()

DELAYS

delay_cycles()
delay_ms()
delay_us()

DISCRETE I/O

get_tris_X()
input_change_X()
input_state()
input_X()
output_bit()
output_drive()
output_float()
output_high()
output_low()
output_toggle()
output_X()
port_X_pullups()
set_mode_X()
set_pulldown()
set_pullup()
set_slow_slew_X()
set_tris_X()

CAPTURE/COMPARE PWM

get_capture()
set_compare_time()
set_power_pwm_override()
set_power_pwm0_duty()
set_power_pwm2_duty()
set_power_pwm4_duty()
set_power_pwm6_duty()
set_pwm_duty()
set_pwm_period()
setup_capture()
setup_ccp1_duty()
setup_ccp1()
setup_ccp2_duty()
setup_ccp2()
setup_ccp3_duty()
setup_ccp3()
setup_ccp4_duty()
setup_ccp4()
setup_ccp5()
setup_ccp6_duty()
setup_compare()
setup_power_pwm_faults()
setup_power_pwm_pins()
setup_power_pwm()

REAL TIME CLOCK

setup_rtc()*
setup_rtc_alarm()*
rtc_read()*
rtc_write()*
rtc_alarm_read()*
rtc_alarm_write()*

LCD

setup_lcd()
lcd_contrast()
lcd_load()
lcd_symbol()

DIRECT MEMORY ACCESS

dma_start()
dma_status()
setup_dma()

PROCESSOR CONTROLS

act_status()
brownout_enable()
clear_interrupt()
disable_interrupts()
enable_interrupts()
ext_int_edge()
goto_address()
interrupt_active()
jump_to_isr()
label_address()
read_bank()
reset_cpu()
restart_cause()
restart_wdt()
setup_act()
setup_counters()
setup_oscillator()
setup_wdt()
sleep_ulpwu()
sleep()
write_bank()

BIT MANIPULATION

_mul()
bit_clear()
bit_first()
bit_last()
bit_set()
bit_test()
make16()
make32()
make8()
rotate_left()
rotate_right()
shift_left()
shift_right()
swap()

RTOS

rtos_await()
rtos_disable()
rtos_enable()
rtos_msg_poll()
rtos_msg_read()
rtos_msg_send()
rtos_overrun()
rtos_run()
rtos_signal()
rtos_stats()
rtos_terminate()
rtos_wait()
rtos_yield()

MEMORY ACCESS

erase_eeprom()
erase_program_eeprom()
erase_program_memory()
read_calibration()
read_configuration_memory()
read_eeprom()
read_extended_ram()
read_external_memory()
read_program_eeprom()
read_program_memory()

read_program_memory8()
read_rom_memory()
setup_external_memory()
write_configuration_memory()
write_eeprom()
write_extended_ram()
write_external_memory()
write_program_eeprom()
write_program_memory()
write_program_memory8()

QUADRATURE ENCODER INTERFACE

setup_qei()*
qei_get_count()*
qei_set_count()*
qei_status()*

STANDARD C CHAR

atol()
isalnum()
isalpha()
isamong()
isamoung()
iscntrl()
isdigit()
isgraph()
islower()
isprint()
ispunct()
isspace()
isupper()
isxdigit()
itoa()
sprintf()
strcat()
strchr()
strcmp()
strcoll()
strcpy()
strncpy()
strerror()
stricmp()
strlen()
strlwr()
strncat()
strncmp()
strncpy()
strpbrk()
strrchr()
strspn()
strstr()
strtod()
strtof()
strtof48()
strtok()
strtol()
strtoul()
strxfrm()
tolower()
toupper()



Project Navigation

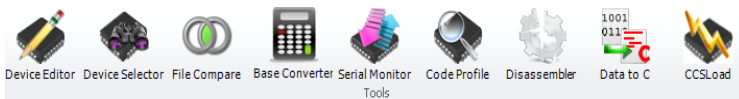
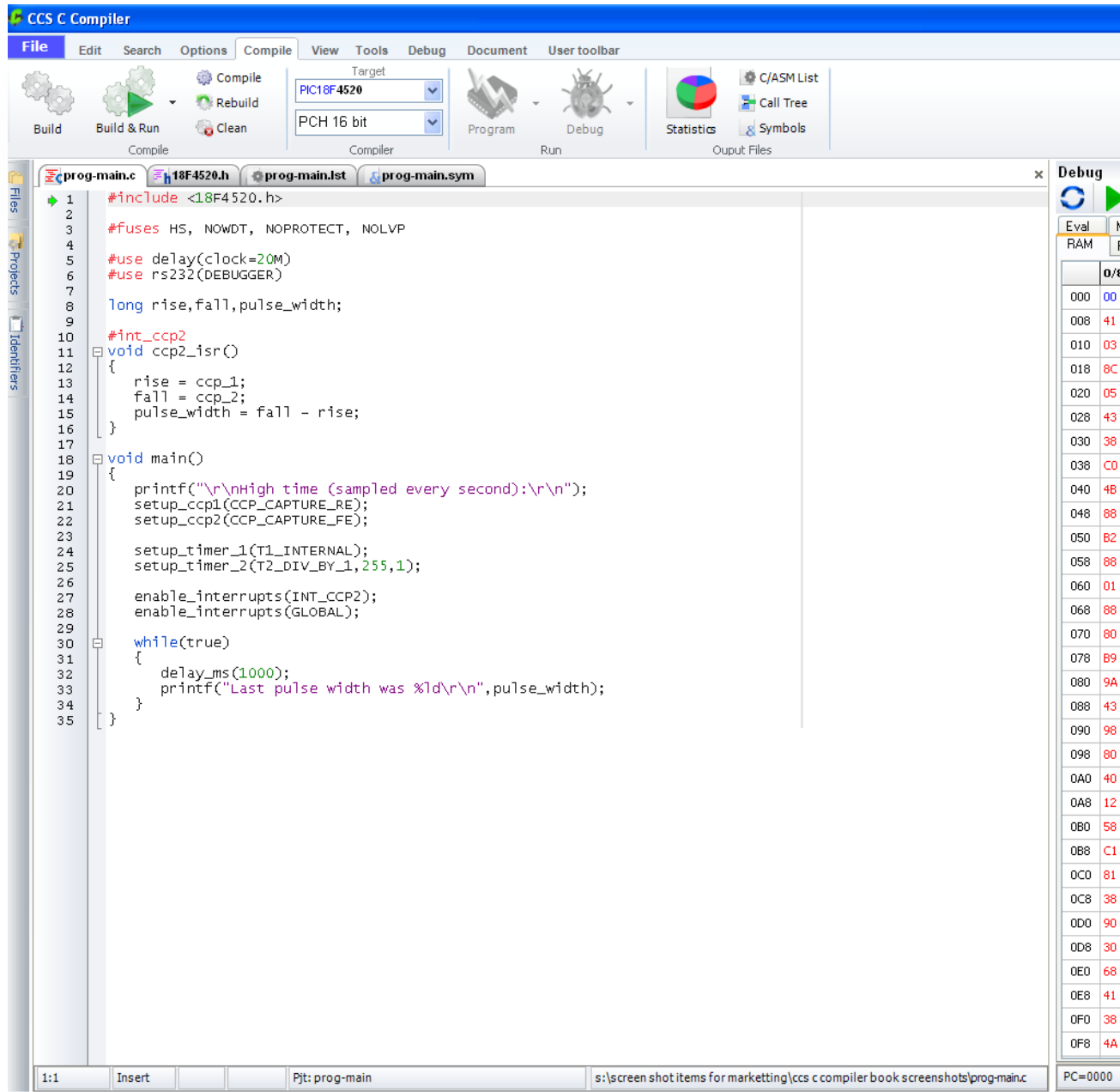
File bar shows all project related files and can quickly open or compile a file.
Identifier bar shows all project functions and identifiers.



Project Wizard

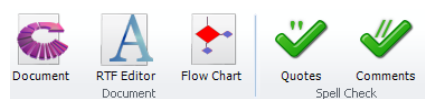
Wizards

Simplifies configuration of drivers and peripherals to start projects quickly. Forms based on interactive questions to aid in set-up of options, such as calculating and showing the timer options based on your clocks.
Included in PCW: CAN Bus, USB, RS-485, and many more



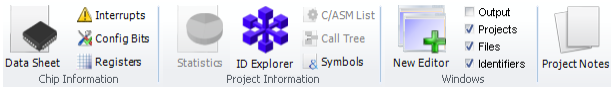
Tools

- Device Selector
- Edit/Add to Device Database
- Generate C Constant
- Declarations from hex/binary
- Special Function Register Reference
- Serial Port Monitor
- File compare for list or source files



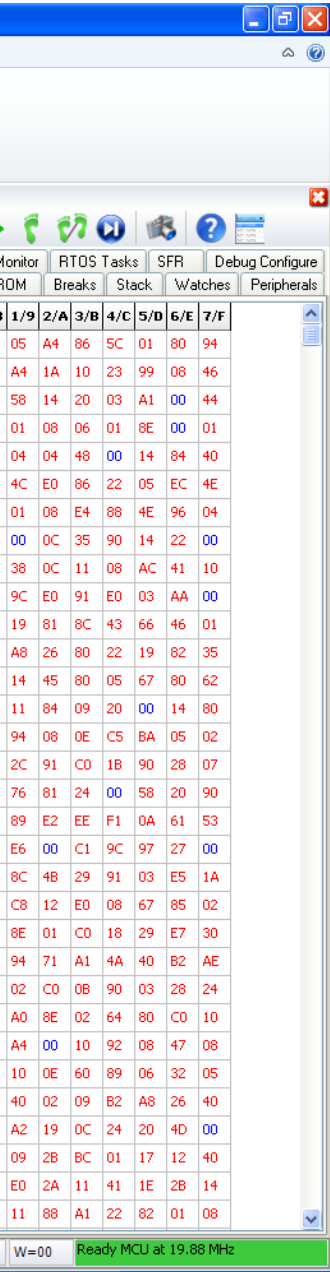
Editor Features

- RTOS--integrated for maximum efficiency and multi-tasking allowed with deterministic scheduling
- Automated C indenting
- Context Sensitive Help
- Color Syntax Highlighting



Special Viewers

Include quick and easy access to data sheets, valid fuses, interrupts for devices, hex file disassembler, .COD file interpreter, and an advanced source/list file compare.



RAM

The RAM window allows the user to view all the memory locations in the device RAM.

C brace matching
Multiple open windows
Technical Support Wizard
Multiple Compilation Unit preprocessor directives
Documentation Generator

Flow Chart Editor
RTF Documentation Generator
Spellchecker
Download Manager

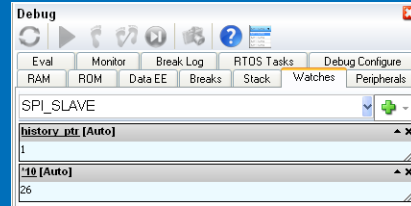
C Aware Real-time Debugger

*Can be used with the CCS ICD and Mach X, and Microchip MPLABX®, REAL ICE™.



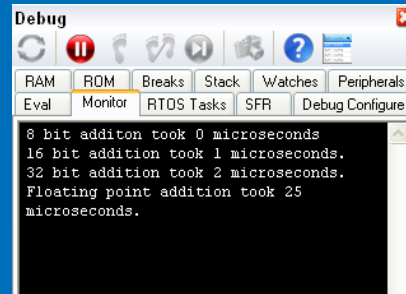
Watches

Full C expressions can be specified. Arrays and structures are understood and shown in natural form. Variables can be modified and break points can be set.



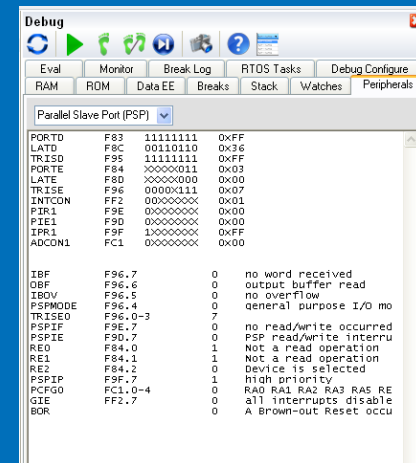
Serial Data

A monitor allows character I/O to and from the target platform. The target platform can printf to this debugger window and getch from it.



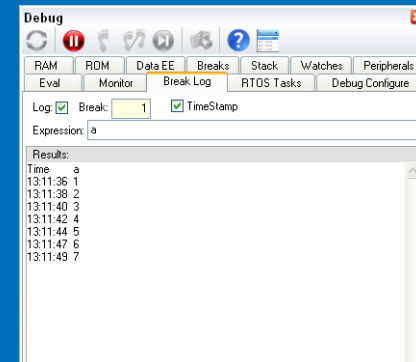
Peripherals

Special function registers are grouped by a function and each bit is fully interpreted in the debugger window.

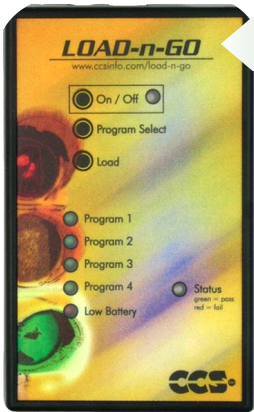


Logging

Configurable to save data each time a specified source line is executed. Set-up profiles can be saved and used with any project. Debugger data can be printed or saved to disk file.



Programming and Debugging Solutions that Meet ALL of Your Requirements!



LOAD-n-GO Handheld Programmer

- Stores up to four firmware images
- Update products in the field
- Enables technicians and engineers to update deployed products without a PC
- Programs devices via ICSP™ even when target system is powered down
- Ability to program PIC32 family

Sku# 53503-814



MACH X Programmer

- Supports all PIC and dsPIC families, MCPxxxx(CAN Bus chips) and PIC32
- In-Circuit programming via a standard ICD connector
- Use as a debugger with use of CCS C through USB
- Program and read HEX and COF files
- User selectable verify voltages (2V to 5.5V) fulfills all requirements on a production programmer required by Microchip's ICSP specifications
- Debugging ability for ISCP devices

Sku# 53500-503

All CCS debuggers and programmers are supported by CCSLOAD, the free CCS Programmer Control Software. CCSLOAD features a Windows user interface with extensive diagnostics and security options and a command line interface that will run on Linux and Window's platforms.



ICD-U64 In-Circuit Programmer/Debugger

- Fast and easy-to-use ICD solution
- Supports all Microchip's PIC® MCUs and dsPIC®DSCs
- Covers all targets that have debug mode when used in conjunction with CCS IDE compilers
- Provides In-Circuit Serial Programming (ICSP) support for all Flash chips
- Powers and connects through USB
- Ability to program PIC32 family

Sku# 53502-852



Prime8 Production ICSP™ Programmer

- Carry out low-cost production programming
- Programs up to 8 devices concurrently
- Operates in stand-alone mode or while connected to a PC
- Supplies up to 200ma at 2-5V to power target devices
- Ability to program PIC32 family

Sku# 53504-830

About CCS

CCS is a leading worldwide supplier of embedded software development tools that enable companies to develop premium products based on Microchip PIC® MCU and dsPIC® DSC devices. Complete proven tool chains from CCS include a code optimizing C compiler, application specific hardware platforms and software development kits. CCS products accelerate development of energy saving industrial automation, wireless and wired communication, automotive, medical device and consumer product applications.

Established in 1996, CCS is a Microchip Premier 3rd Party Partner. For more information, please visit www.ccsinfo.com.



sales@ccsinfo.com
support@ccsinfo.com

262-522-6500 x35
262-522-6500 x32

Development Kits from CCS - Enable a design advantage!



CCS Development kits combine the powerful and optimized CCS C Compiler, an ICD-U64 In-Circuit Programmer/Debugger, application specific prototyping boards, and all required hardware accessories.

Kit Name	Push Button	3 LEDs	POTS	RS-232	I/O pins		IDE Compiler	Special Features
					Total	Analog		
Capacitive Touch		1		1			PCW	PIC16LF727 with mTouch™ Sensing Solution. Equipped with *Tag Connect footprint for ICSP programming.
CCS Wireless - Ember ZigBee™ Edition					15	4	PCWH	Full communication protocols with the Ember-ZigBee™ stack using the powerful Ember EM260 chip.
DSP Starter	✓	✓	1	1	10	1	PCDIDE	Real ICE™ connector & a header to access the available GPIO, Runs at 30 MIPS
DSP Analog	✓	✓	2	1	21	8	PCWHD	Includes a Texas Instruments TLV320AIC23B audio codec
Embedded Internet	✓	2	2	1	6	1	PCWH	Realtek 8019A NE2000 Compatible NIC IC, with Ethernet jack
Embedded Ethernet	✓	✓	1	1	30	12	PCWH	Header for I/O to PIC18F4620, LCD, Serial EEPROM, MMC card reader
3.3V Ethernet Controller	✓	✓	1	1	20	2	PCWH	PIC18F67J60, prototyping board operating at 10-base T speeds
Robotics					10	2	PCW	WTS701 Text-to-Speech Converter with Speaker, Proximity Detection, Infrared Detection, Ball Bearing Servo Motors & Expansion Port
Embedded Serial Busses		✓			30	7	PCW	Has two nodes and shares common I/O, memory and sensor components between two PIC16 MCUs
CAN Bus		9	3	3	30	10	PCWH	Includes a PIC18F4580, a PIC16F876A connected to an MCP2515 CAN peripheral, and two MCP25050 CAN expanders.
CAN Bus 24		9	3	2	30	8	PCWHD	Includes a PIC24HJ56GP610 with two on-chip ECAN controllers, a dsPIC30F4012 connected to an MCP2515 CAN peripheral, and two MCP25050 CAN expanders.
USB	✓	✓	1	2	26	11	PCWH	PIC18F4550 with an on-chip USB controller peripheral
USB Master	2	✓	1	2	14	4	PCWH	Hosts a Vinculum™ VNC1L chip
RFID		4		1			PCW	Short range RFID antenna connected to an external RFID transceiver IC
USB for PIC24	✓	✓	1	2	30	5	PCWHD	PIC24FJ256GB206 with an on-chip USB controller peripheral
PIC12F675	✓	✓	2	1	6	4	PCW	Board includes the 14-pin part for ICSP & debugging at the C level
PIC12F683	✓	✓	2	1	6	4	PCW	Similar to PIC12F675 with twice the RMA & EEPROM, 3 timers & Capture/Compare/PWM module
PIC16F877A	✓	✓	1	1	30	7	PCWH	Basic features for quick and easy learning
PIC16F887	✓	✓	1	1	30	12	PCWH	Basic features of 877A family with additional I/O
101:C on PIC16F818	✓	6			4	2		Complete C learning system with powerful and easy-to-use features on the PIC16F818.
PIC16F1937	✓	✓	1	1			PCW	Includes a 93LC56 serial EEPROM, DS1631 digital thermometer and NJU6355 real-time clock.
PIC18F4520	✓	✓	1	1	30	11	PCW	Basic features that require more RMA and Data EEPROM space
PIC18F6722	✓	✓	1	2	48	11	PCW	RS-232 Level Converter connected to the C6/C7 UART and G1G2 UART
PIC18F8722	✓	✓	1	2	29	13	PCW	External Flash & RAM
PIC18F45K22	✓	✓	1	1	30	11	PCWH	Includes 93LC56 serial EEPROM chip, DS1631 digital thermometer chip, and NJU6355 real-time clock IC with attached 32.768kHz crystal
PIC18F67J10	✓	✓	1	2	48	10	PCWH	Basic features for 3.3V applications
PIC24F	✓	✓	1	2	48	16	PCDIDE	Runs at 16 MIPS
PIC24H	✓	✓	1	2	48	18	PCDIDE	Runs at 40 MIPS-Low drive current
ACE Kit	2	5	2	2			PCW	PIC16F877A with connectors and expanders