Triscend Development Tools Update

Triscend E5 Support

The Triscend E5 family of Configurable System-on-Chip (CSoC) devices is based on a performance accelerated 8-bit 8051 microcontroller. The E5 Family consists of several devices which vary in the amount of on-chip programmable logic, RAM, and I/O. The E5 is ideal for embedded systems applications that demand both fast time-to-market and high levels of customization.

Software development tools (assembler, ANSI C Compiler, and target debugger) from Keil Software fully support the E5 and provide a platform for programming and debugging your real-time, in-system target hardware.

Configurable System-on-Chip (CSoC)

Designers may configure E5 CSoC chips using the Triscend **FastChip** Development System. **FastChip** software customizes the processor's peripherals using a *drag and drop* methodology which allows you to create your own 8051 derivatives on-demand. Simply select the desired peripheral from the library and *drag* it into place. **FastChip** makes the rest of the process automatic so you can begin code development without worrying about the implementation details of the peripheral set.

JTAG Debugging with µVision2

The Triscend E5 family supports target-level debugging via the built-in JTAG interface. Using the Triscend JTAG driver with the Keil *µ*Vision2 Debugger, you may test application programs running on your actual target hardware.

The JTAG interface allows you to download program code, set breakpoints, watch memory locations, and single-step through your C or assembly programs. Since the JTAG interface works with the Keil debugger, you can instantly get started writing and testing embedded programs for the Triscend E5 family. The JTAG debugger gives you full access to the Triscend E5 peripherals.

This Newsletter Contains Information About:

KEIL™ SOFTWARE

µVISION 2

C51

Debug Triscend Applications
Using the µVision2 Debugger

Sophisticated µVision2 IDE
Makes Getting Started Easy

FastChip Design Flow (Part 1)

The Triscend E5 is a Configurable System-on-Chip (CSoC) that you can easily customize to meet your target requirements. Using the FastChip software, you specify how to use the standard on-chip peripherals and select additional peripherals to include from the module library.

The following configuration example shows just how easy this is. To begin, start FastChip and create a new project.

Select and Configure Standard Peripherals

Each E5 device has dedicated resources (peripherals) that are available. These include the default peripherals found on a standard 805l microcontroller.

The first step in configuring your FastChip project is to select and configure the on-chip peripherals for your application. FastChip then writes the code necessary to initialize the peripherals you selected.

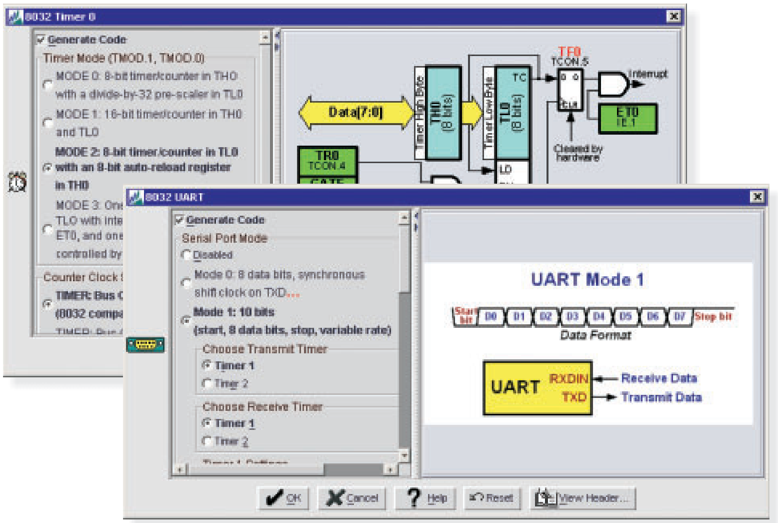
Drag and Drop Custom Peripherals

FastChip comes with a large module library of additional peripherals you can drag and drop directly into your project. Choose from I/O Ports, Pulse Width Modulators, SPI, or whatever your target application requires.

Configure and Connect Peripherals

After you select the dedicated resources and add custom peripherals to your project, you must configure each peripheral and connect it to the E5.

Each peripheral from the module library provides a dialog where you may specify the names to use for I/O (connections) and define the characteristics (properties).



8032 Timer 0

☑ Generate Code

Timer Mode (TMOD.1, TMOD.0)

○ MODE 0: 8-bit timer/counter in TH0 with a divide-by-32 pre-scaler in TL0

○ MODE 1: 16-bit timer/counter in TH0 and TL0

● MODE 2: 8-bit timer/counter in TL0 with an 8-bit auto-reload register in TH0

○ MODE 3: One TL0 with inte ET0, and one controlled by

Counter Clock S

● TIMER: Bus C (8032 compa

8032 UART

☑ Generate Code

Serial Port Mode

○ Disabled

○ Mode 0: 8 data bits, synchronous shift clock on TXD...

● Mode 1: 10 bits (start, 8 data bits, stop, variable rate)

Choose Transmit Timer

● Timer 1

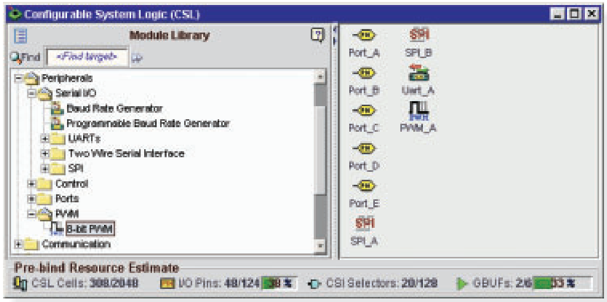
○ Timer 2

Choose Receive Timer

● Timer 1

○ Timer 2

Step 1. *Select the standard 805l peripherals for your target E5 device.*



Configurable System Logic (CSL)

Module Library

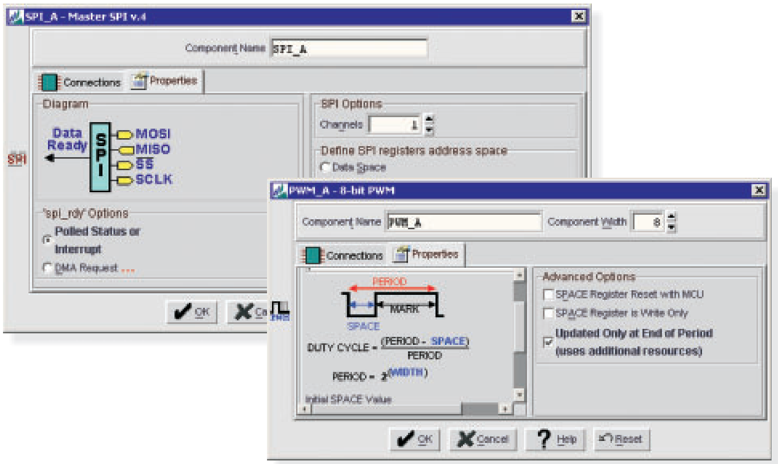
Find «Find target»

- Peripherals
  - Serial I/O
    - Baud Rate Generator
    - Programmable Baud Rate Generator
    - UARTs
    - Two Wire Serial Interface
    - SPI
  - Control
  - Ports
  - PWM
    - 8-bit PWM
- Communication

Pre-bind Resource Estimate

CSL Cells: 308/2048 I/O Pins: 48/124 38 % CSI Selectors: 20/128 GBUFs: 2/6 33 %

Step 2. *Drag and drop customized peripherals into your design.*



SPI_A - Master SPI v.4

Component Name SPI_A

Connections Properties

SPI Options

Channels 1

Define SPI registers address space

○ Data Space

'spi_rdy' Options

● Polled Status or Interrupt

○ DMA Request ...

PWM_A - 8-bit PWM

Component Name PWM_A Component Width 8

Connections Properties

Advanced Options

□ SPACE Register Reset with MCU

□ SPACE Register is Write Only

☑ Updated Only at End of Period (uses additional resources)

DUTY CYCLE = (PERIOD - SPACE) / PERIOD

PERIOD = 2^(WIDTH)

Initial SPACE Value

Step 3. *Connect peripherals, assign names, and select memory spaces.*

Generate Source Code

After selecting and configuring the on-chip peripherals in your FastChip project, you may generate C or assembler source code to initialize them. The source code created by FastChip is fully documented and even includes definitions of the bits of each SFR (Special Function Register) that are used.

Code that is generated is fully compatible with the Keil 8051 C compiler and may easily be included in your µVision2 projects.

Assign I/O Pins

Most custom peripherals in your FastChip project require some form of input and output. Those that interact with external signals require that you connect pins on the E5 CSoC to the internal signals of the peripheral.

The I/O Editor shows unassigned I/O pads of custom peripherals in the left window and pins of the E5 device in the right window. Using the editor, FastChip allows you to simply drag and drop the peripheral's signal name onto the desired E5 pin to make that connection.

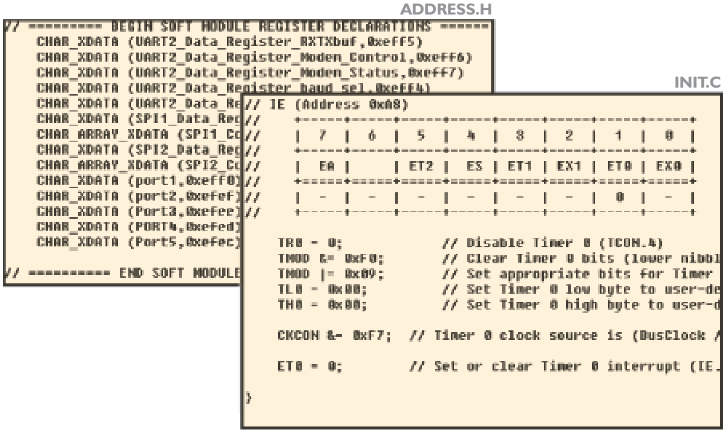
As signals are connected, they are annotated using the symbols shown in the legend at the bottom left.

Bind Your Design

Before you can download your design to the E5 device, you must bind the design to the device's resources. The binding process determines the configuration information FastChip will download to the E5.

When you bind your project, you may specify the amount of effort FastChip expends.

During the bind process, FastChip displays status, errors, warnings, and a summary. If there are problems that prevent the design from being bound, you may correct them and re-bind the project.



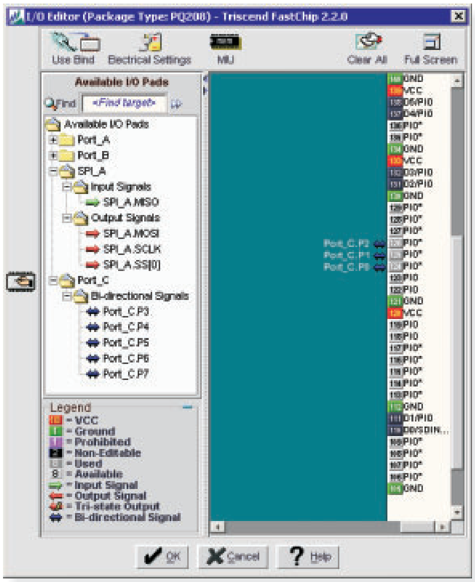
ADDRESS.H

```
// ========== BEGIN SOFT MODULE REGISTER DECLARATIONS ======
    CHAR_XDATA (UART2_Data_Register_RXTXbuf,0xeFF5)
    CHAR_XDATA (UART2_Data_Register_Moden_Control,0xeFF6)
    CHAR_XDATA (UART2_Data_Register_Moden_Status,0xeff7)
    CHAR_XDATA (UART2_Data_Register_baud_sel,0xeFF4)
    CHAR_XDATA (UART2_Data_Re
    CHAR_XDATA (SPI1_Data_Reg
    CHAR_ARRAY_XDATA (SPI1_Co
    CHAR_XDATA (SPI2_Data_Reg
    CHAR_ARRAY_XDATA (SPI2_Co
    CHAR_XDATA (port1,0xeff0)
    CHAR_XDATA (port2,0xefef)
    CHAR_XDATA (Port3,0xefee)
    CHAR_XDATA (PORT4,0xefed)
    CHAR_XDATA (Port5,0xefec)

// =========== END SOFT MODULE
```

INIT.C

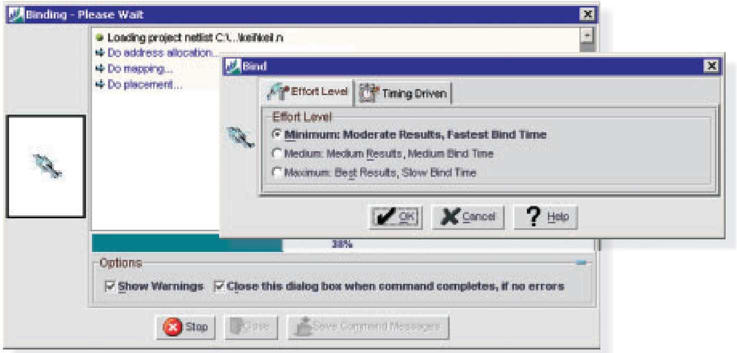
```
// IE (Address 0xA8)
//     +------+------+------+------+------+------+------+------+
//     |  7   |  6   |  5   |  4   |  3   |  2   |  1   |  0   |
//     +------+------+------+------+------+------+------+------+
//     |  EA  |      | ET2  |  ES  | ET1  | EX1  | ET0  | EX0  |
//     +======+======+======+======+======+======+======+======+
//     |  -   |  -   |  -   |  -   |  -   |  -   |  0   |  -   |
//     +------+------+------+------+------+------+------+------+

    TR0 = 0;              // Disable Timer 0 (TCON.4)
    TMOD &= 0xF0;         // Clear Timer 0 bits (lower nibbl
    TMOD |= 0x09;         // Set appropriate bits for Timer
    TL0 = 0x00;           // Set Timer 0 low byte to user-de
    TH0 = 0x00;           // Set Timer 0 high byte to user-d

    CKCON &= 0xF7;   // Timer 0 clock source is (BusClock /

    ET0 = 0;         // Set or clear Timer 0 interrupt (IE.

}
```

Step 4. *FastChip automatically generates initialization source code.*



I/O Editor (Package Type: PQ208) - Triscend FastChip 2.2.0

Step 5. *Assign I/O Pins using the I/O Editor.*



Binding - Please Wait

Loading project netlist C:\...\keil\keil.n
Do address allocation...
Do mapping...
Do placement...

Bind

Effort Level

● Minimum: Moderate Results, Fastest Bind Time
○ Medium: Medium Results, Medium Bind Time
○ Maximum: Best Results, Slow Bind Time

38%

Options

☑ Show Warnings ☑ Close this dialog box when command completes, if no errors

Step 6. *Bind your design to the E5 device.*

Combine Logic and Firmware

When you are finished with your design, FastChip combines the logic and firmware and downloads them to your target system. You must specify the path of the Target CSoC memory configuration, the path of the CSL configuration file generated during the Bind Process, and the Intel HEX file (firmware) generated by the Keil tools.

Before downloading, you may configure the Target Memory Device, the CSI Bus Clock source and speed, and the type of security.

The download process is automatic. All the necessary files, logic, and firmware are transferred to the target via the on-chip JTAG interface.

Your Custom 8051 is Ready to Use

And, voila! Your customized 8051 CSoC system is now ready to use.

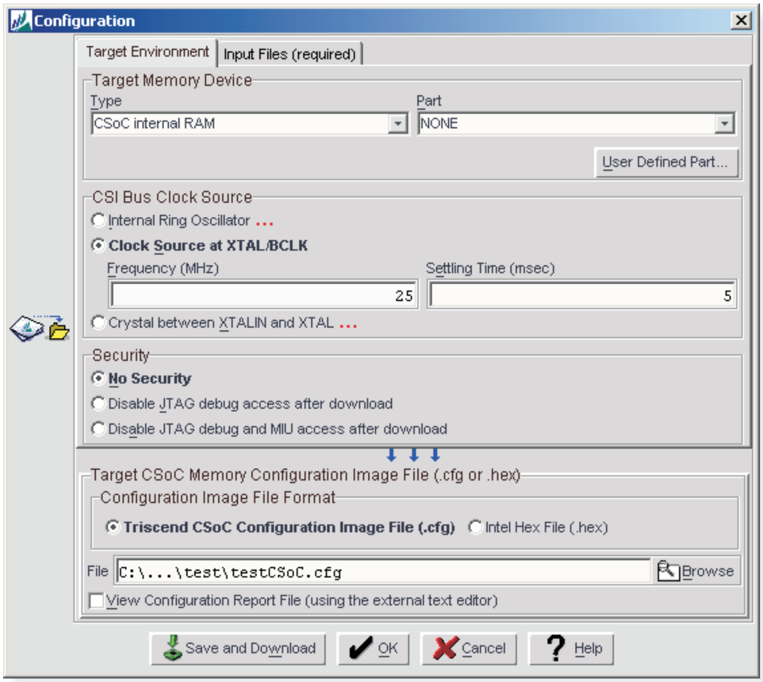
Debug and Test

Debugging and testing an embedded CSoC application is extremely easy due to tight integration with the Keil Software μVision2 Debugger. Using the Keil tools, you may immediately start debugging your target application on the actual hardware using the Triscend Debugger Driver (for μVision2) and the on-chip JTAG interface.

Combining the ease and simplicity of the FastChip design software with the flexibility of the Keil μVision2 IDE provides a robust CSoC development environment.

How To Get FastChip

Visit **www.triscend.com** to download a fully-functional evaluation version of FastChip and get started designing your CSoC-based system today.


Configuration

Target Environment | Input Files (required)

Target Memory Device
Type: CSoC internal RAM
Part: NONE
User Defined Part...

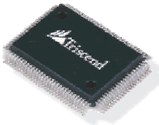
CSI Bus Clock Source
○ Internal Ring Oscillator ...
● Clock Source at XTAL/BCLK
Frequency (MHz): 25
Settling Time (msec): 5
○ Crystal between XTALIN and XTAL ...

Security
● No Security
○ Disable JTAG debug access after download
○ Disable JTAG debug and MIU access after download

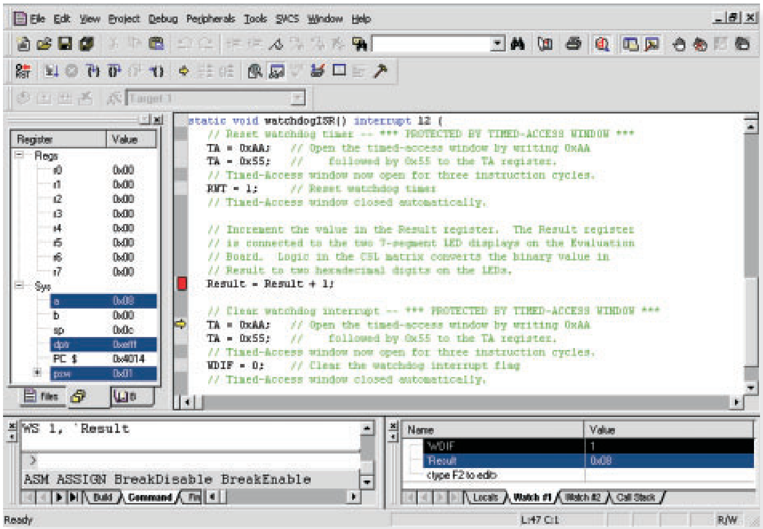
Target CSoC Memory Configuration Image File (.cfg or .hex)
Configuration Image File Format
● Triscend CSoC Configuration Image File (.cfg) ○ Intel Hex File (.hex)
File: C:\...\test\testCSoC.cfg
Browse
□ View Configuration Report File (using the external text editor)

Save and Download | OK | Cancel | Help

***Step 7**. Combine the configuration logic and firmware and download.*



***Step 8**. Your customized 8051 is ready.*


File Edit View Project Debug Peripherals Tools SVCS Window Help
Target 1
Files

Register	Value
Regs	
r0	0x00
r1	0x00
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00
Sys	
a	0x08
b	0x00
sp	0x0c
dptr	0xeff
PC $	0x4014
psw	0x01

```
static void watchdogISR() interrupt 12 {
   // Reset watchdog timer -- *** PROTECTED BY TIMED-ACCESS WINDOW ***
   TA = 0xAA;    // Open the timed-access window by writing 0xAA
   TA = 0x55;    //    followed by 0x55 to the TA register.
   // Timed-Access window now open for three instruction cycles.
   RWT = 1;      // Reset watchdog timer
   // Timed-Access window closed automatically.

   // Increment the value in the Result register.  The Result register
   // is connected to the two 7-segment LED displays on the Evaluation
   // Board.  Logic in the CSL matrix converts the binary value in
   // Result to two hexadecimal digits on the LEDs.
   Result = Result + 1;

   // Clear watchdog interrupt -- *** PROTECTED BY TIMED-ACCESS WINDOW ***
   TA = 0xAA;    // Open the timed-access window by writing 0xAA
   TA = 0x55;    //    followed by 0x55 to the TA register.
   // Timed-Access window now open for three instruction cycles.
   WDIF = 0;     // Clear the watchdog interrupt flag
   // Timed-Access window closed automatically.
```

```
WS 1, `Result
>
ASM ASSIGN BreakDisable BreakEnable
```

Name	Value
WDIF	1
Result	0x08
<type F2 to edit>	

***Step 9**. Debug and test your target application.*

Getting Started

The best way to start your own CSoC project is to get the Triscend E5 Development Kit. This kit includes the E5 evaluation board that is designed to help you create working programs using the E5 family of devices. All features of the board are fully supported by FastChip and μVision2.

The E5 Development Kit provides a flexible and powerful platform for quickly creating, developing, and debugging embedded system designs based on the 8-bit E5 family of CSoC devices. It features the E5 evaluation board, the E5 base board, a power supply, and a parallel cable for download and debug.

E5 Development Kit Features

- Triscend TE520 CSoC Device
- 128 KByte Flash Memory
- Serial Configuration EEPROM Socket
- 40 MHz Oscillator
- JTAG Debug and Download
- Two 7-segment Displays
- RS-232 Serial Channel

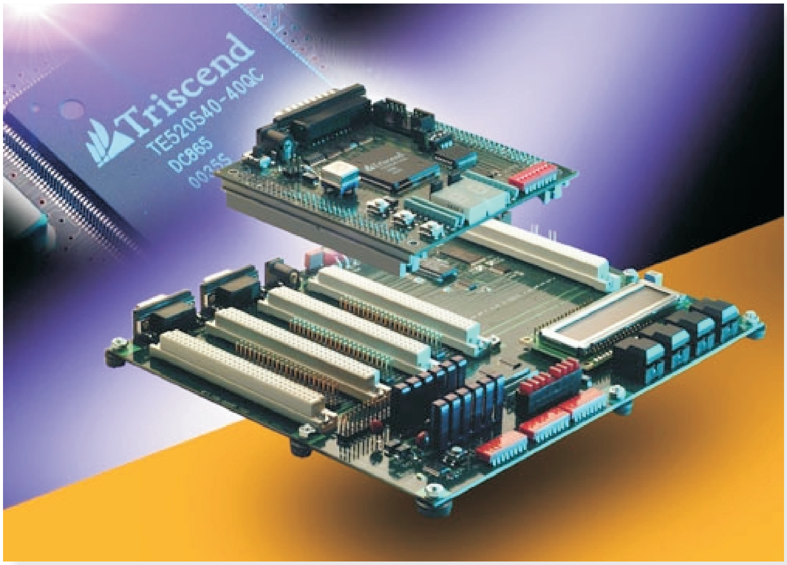
This complete kit is available from Triscend. For more details, visit **www.triscend.com**.

Technical Support

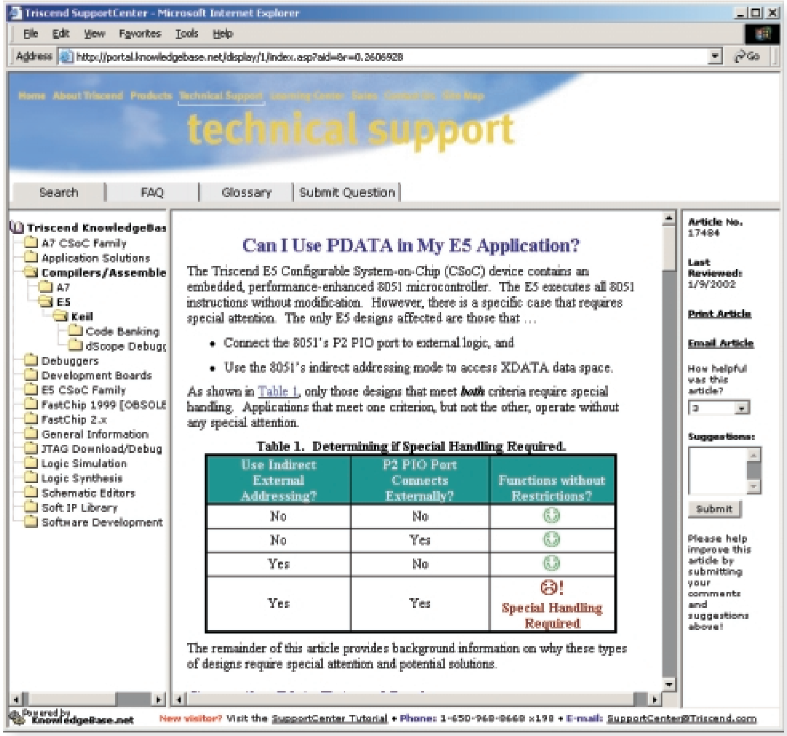
Both Keil and Triscend provide world-class technical support to help answer your development questions.

Triscend provides web-based technical support that is available 24 hours a day for. Their web site includes FAQ's as well as an indexed, searchable knowledge base of technical articles.

Keil Software offers a knowledgebase with over 1500 articles, dozens of application notes, download files, and a discussion forum you may utilize to answer your technical questions. For more information, visit **www.keil.com/support**.



The Triscend E5 Development Kit includes everything you need to get started.



Can I Use PDATA in My E5 Application?

The Triscend E5 Configurable System-on-Chip (CSoC) device contains an embedded, performance-enhanced 8051 microcontroller. The E5 executes all 8051 instructions without modification. However, there is a specific case that requires special attention. The only E5 designs affected are those that …

- Connect the 8051's P2 PIO port to external logic, and
- Use the 8051's indirect addressing mode to access XDATA data space.

As shown in Table 1, only those designs that meet ***both*** criteria require special handling. Applications that meet one criterion, but not the other, operate without any special attention.

Table 1. Determining if Special Handling Required.

Use Indirect External Addressing?	P2 PIO Port Connects Externally?	Functions without Restrictions?
No	No	☺
No	Yes	☺
Yes	No	☺
Yes	Yes	☹! Special Handling Required

The remainder of this article provides background information on why these types of designs require special attention and potential solutions.

Article No. 17484

Last Reviewed: 1/9/2002

How helpful was this article? 3

Suggestions:

Please help improve this article by submitting your comments and suggestions above!

New visitor? Visit the SupportCenter Tutorial • Phone: 1-650-968-8668 x198 • E-mail: SupportCenter@Triscend.com

*Technical Support is available 24/7 from Triscend (**www.triscend.com**) and from Keil Software (**www.keil.com/support**).*

*µ*Vision2 is an IDE that brings together all aspects of your embedded application.

Starting a Project

When creating a new project in *µ*Vision2, you must first select the folder and filename. *µ*Vision2 then prompts you to select the device you will use. The Keil development tools for the 8051 support all 8051 variants including the Triscend E5 Devices.

Organizing Project Settings

*µ*Vision2 projects are composed of one or more targets, groups, and files.

Targets define the chip to use as well as all assembler, compiler, linker, and debugger settings for an executable. A project may have more than one target. For example, you can have a target for debugging and another target for your release software.

Groups provide you with a way to separate and organize the source files in a project. When adding files to the project, you may add all files to a single group or you may create groups for startup code, documents, source, and header files.

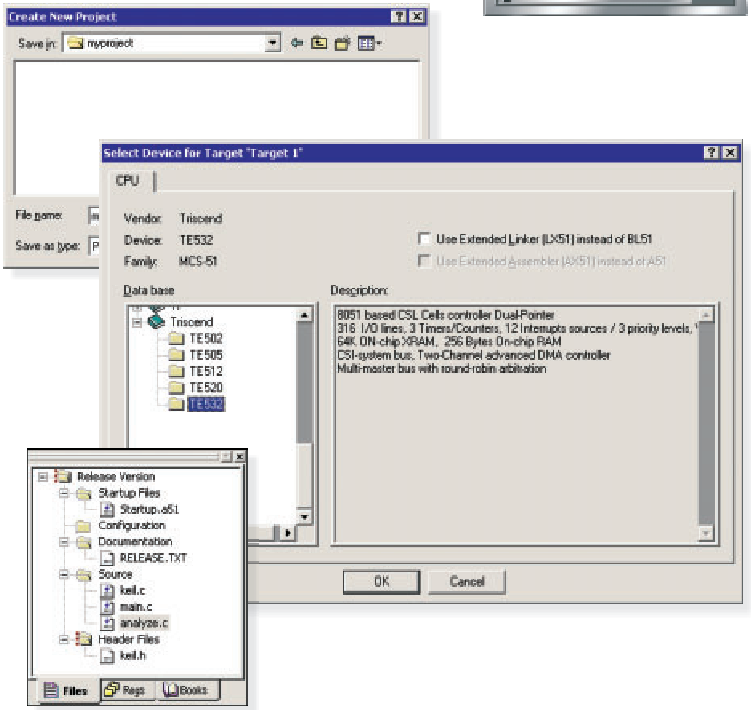
FastChip generates C header and source files you may include in your *µ*Vision2 project to initialize the on-chip peripherals of your Triscend device.

Setting Tool Options

Most tool options are automatically set when you select the device you use. Other options like memory size, folder names, source browser settings, and output controls may be configured from the Project Options Dialog.

Building Your Application

After adding files and setting options, *µ*Vision2 allows you to interactively build and correct syntax errors in your project. Double-click on an error to edit the corresponding line in your program source.



Create New Project

Save in: myproject

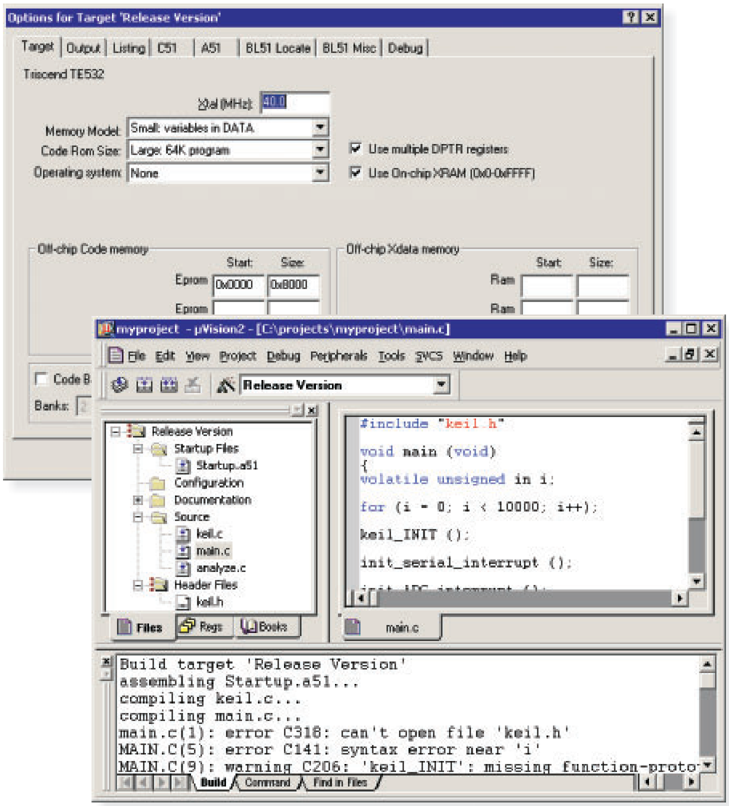
Select Device for Target 'Target 1'

Vendor: Triscend
Device: TE532
Family: MCS-51

□ Use Extended Linker (LX51) instead of BL51
□ Use Extended Assembler (AX51) instead of A51

Data base: Triscend — TE502, TE505, TE512, TE520, TE532

Description:
8051 based CSL Cells controller Dual-Pointer
316 I/O lines, 3 Timers/Counters, 12 Interrupts sources / 3 priority levels,
64K ON-chip XRAM, 256 Bytes On-chip RAM
CSI-system bus, Two-Channel advanced DMA controller
Multi-master bus with round-robin arbitration



Options for Target 'Release Version'

Target | Output | Listing | C51 | A51 | BL51 Locate | BL51 Misc | Debug

Triscend TE532

Xtal (MHz): 40.0
Memory Model: Small: variables in DATA
Code Rom Size: Large: 64K program
Operating system: None
☑ Use multiple DPTR registers
☑ Use On-chip XRAM (0x0-0xFFFF)

Off-chip Code memory — Eprom Start: 0x0000 Size: 0x8000

myproject - µVision2 - [C:\projects\myproject\main.c]

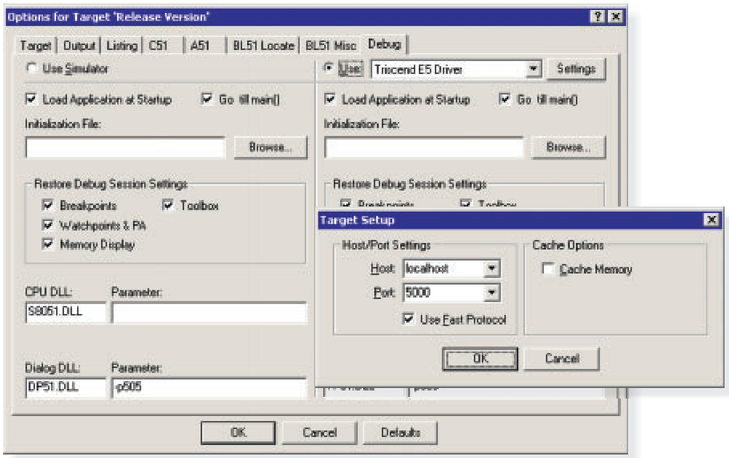
```
#include "keil.h"

void main (void)
{
volatile unsigned in i;

for (i = 0; i < 10000; i++);

keil_INIT ();

init_serial_interrupt ();
```

```
Build target 'Release Version'
assembling Startup.a51...
compiling keil.c...
compiling main.c...
main.c(1): error C318: can't open file 'keil.h'
MAIN.C(5): error C141: syntax error near 'i'
MAIN.C(9): warning C206: 'keil_INIT': missing function-proto
```

Debugging E5 Programs

The *µ*Vision2 Debugger and the Triscend E5 Debugger Driver allow you to test your embedded CSoC applications running on target hardware. When you use the E5 Driver, your target program is downloaded to your target system using the on-chip JTAG interface.

You must select the Triscend Driver as your debugger interface when you configure the debugger. Typically, all driver options are set automatically.



Options for Target 'Release Version'

Tabs: Target | Output | Listing | C51 | A51 | BL51 Locate | BL51 Misc | Debug

○ Use Simulator
● Use: Triscend E5 Driver

Left side:
☑ Load Application at Startup ☑ Go till main()
Initialization File:
Restore Debug Session Settings: ☑ Breakpoints ☑ Toolbox ☑ Watchpoints & PA ☑ Memory Display
CPU DLL: S8051.DLL Parameter:
Dialog DLL: DP51.DLL Parameter: -p505

Right side:
☑ Load Application at Startup ☑ Go till main()
Initialization File:

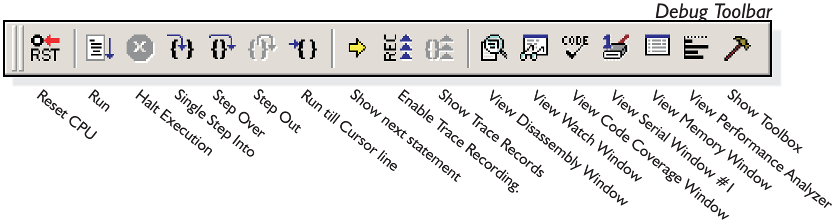
Target Setup
Host/Port Settings: Host: localhost Port: 5000 ☑ Use Fast Protocol
Cache Options: ☐ Cache Memory

Configure the Debugger in the Project Options Dialog.

Executing Code

You may use the buttons on the debug toolbar to step through your application program. The **Run** button executes code until a breakpoint is reached while the **Halt Execution** button stops program execution immediately. **Single Step Into** executes one line of assembly or C source code stepping into a functions.

Debug Toolbar

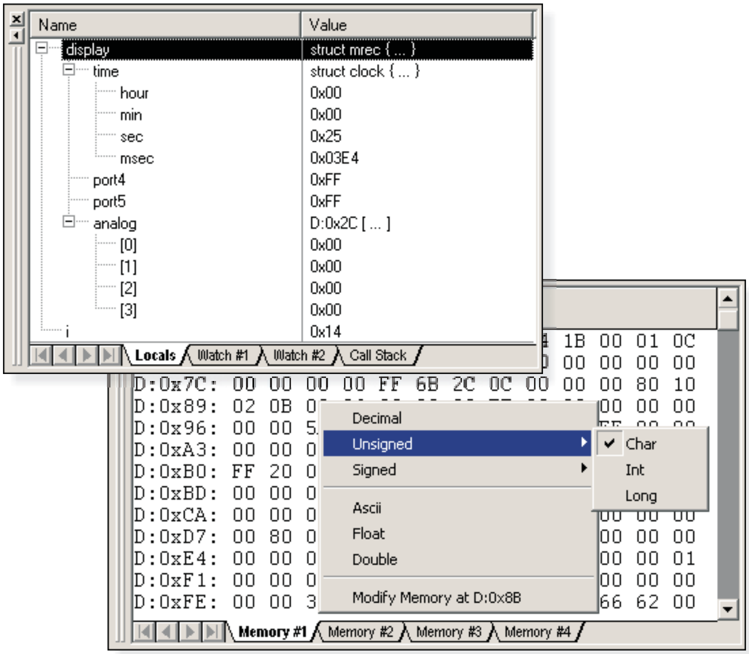


Watching Variables

The *µ*Vision2 Debugger allows you to watch the local variables of each C function. Two user-defined watch windows allow you to specify and watch special program variables. Viewing complex structures, unions, and arrays is supported. Elements may be expanded or collapsed as necessary. Simply select a variable to change its value while debugging.

Viewing Memory

Four memory windows allow you to watch different memory areas used by your program code. You may watch a fixed address or enter a symbol name in the Watch Window Address box.



Name	Value
display	struct mrec { ... }
time	struct clock { ... }
hour	0x00
min	0x00
sec	0x25
msec	0x03E4
port4	0xFF
port5	0xFF
analog	D:0x2C [...]
[0]	0x00
[1]	0x00
[2]	0x00
[3]	0x00
i	0x14

```
                                               4  1B  00  01  0C
                                               0  00  00  00  00
D:0x7C:  00  00  00  00  FF  6B  2C  0C  00  00  00  80  10
D:0x89:  02  0B  0                               00  00  00
D:0x96:  00  00  5.
D:0xA3:  00  00  0
D:0xB0:  FF  20  0
D:0xBD:  00  00  0
D:0xCA:  00  00  0
D:0xD7:  00  80  0                               00  00  00
D:0xE4:  00  00  0                               00  00  01
D:0xF1:  00  00  0                               00  00  00
D:0xFE:  00  00  3                               66  62  00
```

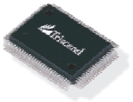
More Details

The *µ*Vision2 Debugger offers features typically available only in high-end emulators. For more information, visit **www.keil.com/uvision2**.

Devices Available from Triscend

Part Number	User I/O	Timers Counters	Interrupt Sources	Watchdog Timer	On-chip XRAM	DMA Controller	CSL Cells	Packages
TE502	92	3	12	✓	8K	✓	256	128LQFP
TE505	124	3	12	✓	16K	✓	512	128LQFP. 208QFP
TE512	188	3	12	✓	32K	✓	1152	128LQFP, 208QFP
TE520	252	3	12	✓	40K	✓	2048	208QFP, 484BGA

Triscend E5 Family Highlights



- Performance accelerated 8051 microcontroller core (10 MIPS at 40 MHz)
- Binary and source code compatible with other 8051 variants
- Stand-alone operation from a single external memory (code + configuration)
- Up to 40 KBytes of on-chip, dedicated system RAM
- Up to 2048 Configurable System Logic (CSL) cells (up to 25,000 **ASIC** gates)
- Power-down and power-management modes (low power mode consumption under 100 μA)
- Compliant to the **CSI Socket** interface, allowing soft peripherals to be used on other CSoC Families
- Two dedicated DMA Channels
- On-chip breakpoint unit provides sophisticated debugging capability
- JTAG debugging using the Keil μVision2 Debugger

KEIL™ SOFTWARE
**1501 10th Street, Suite 110
Plano, TX 75074**

**Phone 800-348-8051
972-312-1107
FAX 972-312-1159**