C8051F04X DEVELOPMENT KIT USER'S GUIDE

1. Kit Contents

The C8051F04x Development Kit contains the following items:

- Two C8051F040 Target Boards
- Serial Adapter (RS232 to Target Board Debug Interface Protocol Converter)
- Silicon Laboratories IDE and Product Information CD-ROM. CD content includes:
  - Silicon Laboratories Integrated Development Environment (IDE)
  - Keil 8051 Development Tools (macro assembler, linker, evaluation 'C' compiler)
  - Installation Utility (SETUP.EXE)
  - Source code examples and register definition files
  - Documentation
- Two AC to DC Power Adapters
- RS232 Serial Cable
- CAN Cable (DB-9 connectors)
- 7" Ribbon Cable
- Quick-start Guide
- C8051F04x Development Kit User's Guide (this document)

2. Hardware Setup

The target board is connected to a PC running the Silicon Laboratories IDE via the Serial Adapter as shown in Figure 1.

1. Connect one end of the RS232 serial cable to a Serial (COM) Port on the PC.
2. Connect the other end of the RS232 serial cable to the DB-9 connector on the Serial Adapter.
3. Connect the Serial Adapter to the JTAG connector on the target board with the 10-pin ribbon cable.
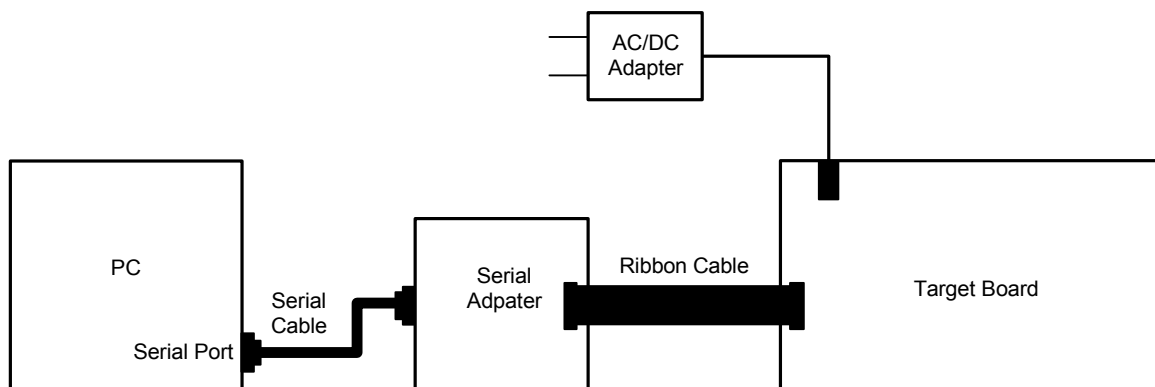4. Connect the AC/DC power adapter to power jack P1 on the target board.

Figure 1. Hardware Setup

> **Note:** The Reset switch on the target board is disabled when the serial adapter is connected to the target board. Use the **Reset** button in the Silicon Laboratories IDE toolbar to reset the target when connected to the Serial Adapter.

C8051F04x-DK

3. Software Setup

The included CD-ROM contains the Silicon Laboratories Integrated Development Environment (IDE), Keil software 8051 tools and additional documentation. Insert the CD-ROM into your PC's CD-ROM drive. An installer will automatically launch, allowing you to install the IDE software or read documentation by clicking buttons on the Installation Panel. If the installer does not automatically start when you insert the CD-ROM, run *autorun.exe* found in the root directory of the CD-ROM. Refer to the *readme.txt* file on the CD-ROM for the latest information regarding known IDE problems and restrictions.

4. Silicon Laboratories Integrated Development Environment

The Silicon Laboratories IDE integrates a source-code editor, source-level debugger and in-system Flash programmer. The use of third-party compilers and assemblers is also supported. This development kit includes the Keil Software A51 macro assembler, BL51 linker and evaluation version C51 'C' compiler. These tools can be used from within the Silicon Laboratories IDE.

4.1. System Requirements

The Silicon Laboratories IDE requirements:

- Pentium-class host PC running Microsoft Windows 95 or later, or Microsoft Windows NT or later.
- One available COM port.
- 64 MB RAM and 40 MB free HD space recommended.

4.2. Assembler and Linker

A full-version Keil A51 macro assembler and BL51 banking linker are included with the development kit and are installed during IDE installation. The complete assembler and linker reference manual can be found on-line under the **Help** menu in the IDE or in the "*SiLabs\MCU\hlp*" directory (A51.pdf).

4.3. Evaluation C51 'C' Compiler

An evaluation version of the Keil C51 'C' compiler is included with the development kit and is installed during IDE installation. The evaluation version of the C51 compiler is the same as the full professional version except code size is limited to 4 kB and the floating point library is not included. The C51 compiler reference manual can be found under the **Help** menu in the IDE or in the "*SiLabs\MCU\hlp*" directory (C51.pdf).

4.4. Using the Keil Software 8051 Tools with the Silicon Laboratories IDE

To perform source-level debugging with the IDE, you must configure the Keil 8051 tools to generate an absolute object file in the OMF-51 format with object extensions and debug records enabled. You may build the OMF-51 absolute object file by calling the Keil 8051 tools at the command line (e.g. batch file or make file) or by using the project manager built into the IDE. The default configuration when using the Silicon Laboratories IDE project manager enables object extension and debug record generation. Refer to Applications Note **AN104 - Integrating Keil 8051 Tools Into the Silicon Laboratories IDE** in the "*SiLabs\MCU\Documentation\Appnotes*" directory on the CD-ROM for additional information on using the Keil 8051 tools with the Silicon Laboratories IDE.

To build an absolute object file using the Silicon Laboratories IDE project manager, you must first create a project. A project consists of a set of files, IDE configuration, debug views, and a target build configuration (list of files and tool configurations used as input to the assembler, compiler, and linker when building an output object file).

The following sections illustrate the steps necessary to manually create a project with one or more source files, build a program and download the program to the target in preparation for debugging. (The IDE will automatically create a single-file project using the currently open and active source file if you select **Build/Make Project** before a project is defined.)

SILICON LABORATORIES

4.4.1. Creating a New Project

1. Select **Project->New Project** to open a new project and reset all configuration settings to default.

2. Select **File->New File** to open an editor window. Create your source file(s) and save the file(s) with a recognized extension, such as .c, .h, or .asm, to enable color syntax highlighting.

3. Right-click on "New Project" in the **Project Window**. Select **Add files to project**. Select files in the file browser and click Open. Continue adding files until all project files have been added.

4. For each of the files in the **Project Window** that you want assembled, compiled and linked into the target build, right-click on the file name and select **Add file to build**. Each file will be assembled or compiled as appropriate (based on file extension) and linked into the build of the absolute object file.

   Note: If a project contains a large number of files, the "Group" feature of the IDE can be used to organize. Right-click on "New Project" in the **Project Window**. Select **Add Groups to project**. Add pre-defined groups or add customized groups. Right-click on the group name and choose **Add file to group**. Select files to be added. Continue adding files until all project files have been added.

4.4.2. Building and Downloading the Program for Debugging

1. Once all source files have been added to the target build, build the project by clicking on the **Build/Make Project** button in the toolbar or selecting **Project->Build/Make Project** from the menu.

   Note**:** After the project has been built the first time, the **Build/Make Project** command will only build the files that have been changed since the previous build. To rebuild all files and project dependencies, click on the **Rebuild All** button in the toolbar or select **Project->Rebuild All** from the menu.

2. C8051F04x family devices use the JTAG debug interface. You must select **JTAG** in the **Options->Debug Interface** menu to enable connection to C8051F04x target devices. Click the **Connect** button in the toolbar or select **Debug->Connect** from the menu to connect to the device.

3. Download the project to the target by clicking the **Download Code** button in the toolbar.

   Note: To enable automatic downloading if the program build is successful select **Enable automatic connect/download after build** in the **Project->Target Build Configuration** dialog. If errors occur during the build process, the IDE will not attempt the download.

4. Save the project when finished with the debug session to preserve the current target build configuration, editor settings and the location of all open debug views. To save the project, select **Project->Save Project As...** from the menu. Create a new name for the project and click on **Save**.

C8051F04x-DK

5. Example Source Code

Example source code and register definition files are provided in the "*SiLabs\MCU\Examples\C8051F04x*" directory during IDE installation. These files may be used as a template for code development. Example applications include a blinking LED example which configures the green LED on the target board to blink at a fixed rate. A Controller Area Network (CAN) application example is also included with the C8051F04x development kit.

5.1. Register Definition Files

Register definition files *C8051F040.inc and C8051F040.h* define all SFR registers and bit-addressable control/status bits. They are installed into the "*SiLabs\MCU\Examples\C8051F04x*" directory during IDE installation. The register and bit names are identical to those used in the C8051F04x data sheet. Both register definition files are also installed in the default search path used by the Keil Software 8051 tools. Therefore, when using the Keil 8051 tools included with the development kit (A51, C51), it is not necessary to copy a register definition file to each project's file directory.

5.2. Blinking LED Example

The example source files *blink.asm* and *blinky.c* show examples of several basic C8051F04x functions. These include; disabling the watchdog timer (WDT), configuring the Port I/O crossbar, configuring a timer for an interrupt routine, initializing the system clock, and configuring a GPIO port. When compiled/assembled and linked this program flashes the green LED on the C8051F040 target board about five times a second using the interrupt handler with a C8051F040 timer.

5.3. Controller Area Network (CAN) Application Example

Configuration and use of the CAN controller is documented in the Bosch CAN User's Guide, located in the "*Documentation*" directory on the CD-ROM. Accessing the CAN controller (i.e., accessing the CAN RAM, CAN registers, and moving data to and from the CAN controller) is documented in Silicon Laboratories' C8051F04x data sheet.

An example CAN application is included in the "*Examples\C8051F04x*" directory. Each C8051F040 target board features a push button (labeled P3.7) and a LED (labeled P1.6). After the two target boards are connected together via the provided CAN bus physical layer (i.e. cable, connectors, and CAN transceiver), the example application sends CAN messages between the two target boards containing the state of the push buttons. In this example, each CAN controller has two of the 32 message objects configured: one to send a control signal based on the state of its target board push button, and one to receive a control signal from the other target to see if it should turn on/off its own LED. When a target board receives a message that the push button on the other target board is depressed, it lights its own LED. When a target board receives a message that the push button on the other target board is not depressed, it turns off its own LED. In this way, the push button on one target board controls the LED on the other target board as a virtual control link via a CAN bus.

5.3.1. Setting-up the Application

1. Connect the target boards together at the CAN DB-9 connectors using the CAN cable provided in the development kit, as shown in Figure 2 on page 5. The correct cable has a male connector on both ends. Take care not to connect the CAN cable to the RS232 DB-9 connector. See Figure 3 on page 6 for the location of the CAN DB-9 connector.

2. Compile and link the can1.c example located in the "*Examples\C8051F04x*" directory on the CD-ROM. Choose one of the target boards as Target Board #1. Connect to Target Board #1 and download the can1 project to the C8051F040, following the steps outlined in Section 4.4 on page 2. Once downloaded, close this project in the IDE and disconnect the EC2 from Target Board #1.

3. Connect the EC2 to the other target board, Target Board #2. Open a new project in the IDE and load can2.c into the C8051F040 device, just as was done in step 2 for Target Board #1. Take care not to load can1.c into both devices. Disconnect the EC2 from this board.

SILICON LABORATORIES

You should now have can1.c loaded into Target Board #1, and can2.c loaded into Target Board #2. The CAN cable should be connected to both boards at the CAN DB-9 connectors.

5.3.2. Running the Application

1. Start the application by resetting the device on each target board. Do this by depressing the RESET push button on each target board. As can1.c executes on Target Board #1, and can2.c runs on Target Board #2, the devices are now nodes on a CAN bus.

2. Pressing the P3.7 push button on Target Board #1 will light the LED on Target Board #2. Likewise, when the push button on Target Board #1 is released, the LED on Target Board #2 will turn off. This will work on either target board.

Once this example is running, you have established a simple CAN network with two nodes. When one C8051F040 device senses the push button on its target board is pressed, it sends a "0x11" in the first byte of a CAN message data field. When the button is released, the first byte of the CAN message data field is "0x00". When a C8051F040 device receives a message, it checks whether this byte has a value of "0x11" or "0x00". When the byte is "0x11", the device turns on its LED by setting P1.6 high. When the byte is "0x00", the device turns off its LED by setting P1.6 low.

You may run the example with the EC2 connected to view CAN registers, and CAN message objects in CAN RAM. While connected to one target board, run the code. Depress the RESET button on the other target board. You may use debug and view features of the Silicon Laboratories IDE and on-chip debug logic. To view the CAN SFRs, click **View->Debug Windows->SFRs->CAN0**. To view message objects in CAN RAM, click **View->Debug Windows->CAN0 Message Registers**. To view SFRs and message registers, the device must be in a halt state to update the debug view windows.

Important Note: To view a Message Object in the CAN Message Registers window, you must set its Message Valid bit to 1 in the Message Object's associated Message Arbitration 2 Register (Bit 15, ARBT2). This can be done in code by configuring the IF1 and IF2 registers to set the associated Message Objects' ARBT2 register. A second method to set this bit is available while viewing the Message Object registers in the IDE CAN0 Message Registers view. Click on, and change, the associated Message Objects' ARBT2 register directly. Working in the background the IDE will set the register for you via the IF1 and IF2 registers.
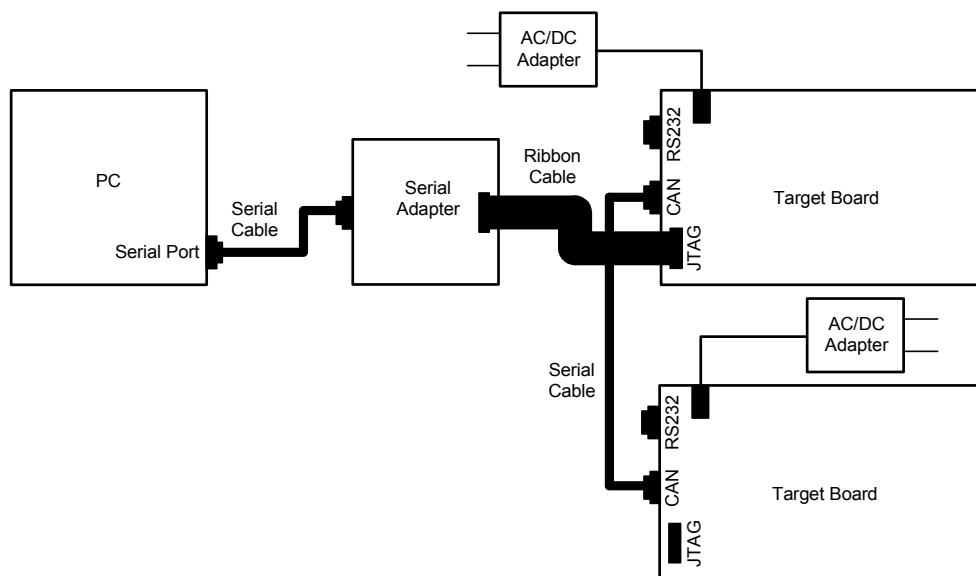

Figure 2. CAN Application Hardware Setup

SILICON LABORATORIES

6. Target Board

The C8051F04x Development Kit includes a target board with a C8051F040 device pre-installed for evaluation and preliminary software development. Numerous input/output (I/O) connections are provided to facilitate prototyping using the target board. Refer to Figure 3 for the locations of the various I/O connectors.

P1	Power connector (accepts input from 7 to 15 VDC unregulated power adapter)
J1	Connects SW2 to P3.7 pin
J3	Connects LED D3 to P1.6 pin
J4	JTAG connector for Serial Adapter interface
J5	DB-9 connector for UART0 RS232 interface
J6	Jumper to connect UART0 TX (P0.0) to DB9
J9	Jumper to connect UART0 RX (P0.1) to DB9
J11	Analog loopback connector
J12-J19	Port 0 - 7 connectors
J20	Analog I/O terminal block
J22	VREF connector
J23	VDD Monitor Disable
J24	96-pin Expansion I/O connector
J25	DB-9 connector for CAN interface

Figure 3. C8051F040 Target Board

SILICON LABORATORIES
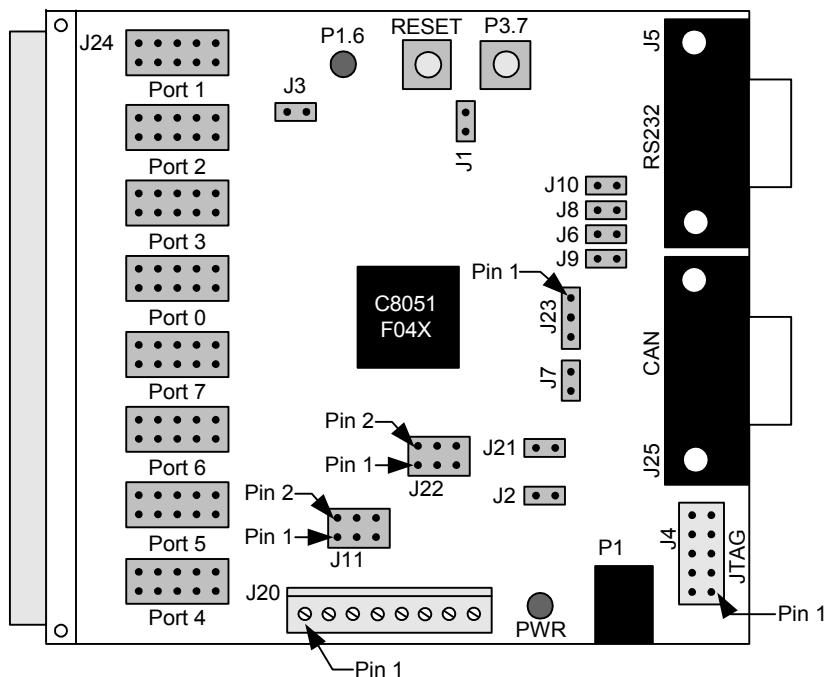
6.1. System Clock Sources

The C8051F040 device installed on the target board features a calibrated programmable internal oscillator which is enabled as the system clock source on reset. After reset, the internal oscillator operates at a frequency of 3.0625 MHz (±2%) by default but may be configured by software to operate at other frequencies. Therefore, in many applications an external oscillator is not required. However, an external 22.1184 MHz crystal is installed on the target board for additional applications. Refer to the C8051F04x data sheet for more information on configuring the system clock source.

6.2. Switches and LEDs

Two switches are provided on the target board. Switch SW1 is connected to the RESET pin of the C8051F040. Pressing SW1 puts the device into its hardware-reset state. Switch SW2 is connected to the C8051F040's general purpose I/O (GPIO) pin through jumpers. Pressing SW2 generates a logic low signal on the port pin. Remove the shorting block from the jumper to disconnect SW2 from the port pins. The port pin signal is also routed to a pin on the J24 I/O connector. See Table 1 for the port pins and jumpers corresponding to each switch.

Two LEDs are also provided on the target board. The red LED labeled PWR is used to indicate a power connection to the target board. The green LED labeled with a port pin name is connected to the C8051F040's GPIO pin through jumpers. Remove the shorting block from the jumper to disconnect the LED from the port pin. The port pin signal is also routed to a pin on the J24 I/O connector. See Table 1 for the port pins and jumpers corresponding to each LED.

Description	I/O	Jumper
SW1	Reset	none
SW2	P3.7	J1
Green LED	P1.6	J3
Red LED	PWR	none

Table 1. Target Board I/O Descriptions

6.3. Target Board JTAG Interface (J4)

The JTAG connector (J4) provides access to the JTAG pins of the C8051F040. It is used to connect the Serial Adapter to the target board for in-circuit debugging and Flash programming. Table 2 shows the JTAG pin definitions.

Pin #	Description
1	+3 VD (+3.3 VDC)
2, 3, 9	GND (Ground)
4	TCK
5	TMS
6	TDO
7	TDI
8, 10	Not Connected

Table 2. JTAG Connector Pin Descriptions

SILICON LABORATORIES

6.4. Serial Interface (J5)

A RS232 transceiver circuit and DB-9 (J5) connector are provided on the target board to facilitate serial connections to UART0 of the C8051F040. The TX, RX, RTS and CTS signals of UART0 may be connected to the DB-9 connector and transceiver by installing shorting blocks on jumpers J6, J8, J9 and J10.

 J6 - Install shorting block to connect UART0 TX (P0.0) to transceiver.
 J9 - Install shorting block to connect UART0 RX (P0.1) to transceiver.
 J8 - Install shorting block to connect UART0 RTS (P4.0) to transceiver.
 J10 - Install shorting block to connect UART0 CTS (P4.1) to transceiver.

6.5. Analog I/O (J11, J20)

Several C8051F040 analog signals are routed to the J20 terminal block and the J11 jumper block. Jumper block J11 provides the ability to connect DAC0 and DAC1 outputs to several different analog inputs by installing a shorting block between a DAC output and an analog input on adjacent pins of J11. Refer to Table 3 for J20 terminal block connections and Table 4 for J11 pin definitions.

Pin #	Description
1	HVAIN+
2	HVAIN-
3	HVREF
4	DAC1
5	AIN0.0
6	AIN0.1
7	VREF0
8	ADND (Analog Ground)

Table 3. J20 Terminal Block Pin Descriptions

Pin #	Description
1	AIN0.0
2	AIN0.1
3	DAC0
4	DAC1
5	AIN0.2
6	AIN0.3

Table 4. J11 Jumper Block Pin Descriptions

SILICON LABORATORIES

6.6. Controller Area Network (CAN) Interface (J25)

A DB-9 (J25) connector is provided to facilitate serial connections to the CAN interface on the C8051040. In addition, when a shorting block is installed on jumper J7, writing a logic 'high' to port pin P4.2 will place the CAN transceiver in low-current standby mode. Also, resistor R12 may be replaced with a higher value to control the slew rate of the CAN_H and CAN_L signals. See the TI SN65HVD230 data sheet for further information. Table 5 listes the pin descriptions for J25.

Pin #	Description
2	CAN_L
7	CAN_H
3, 6	GND (Ground)
1, 4, 5, 8, 9	Not Connected

Table 5. CAN Connector Pin Descriptions

6.7. PORT I/O Connectors (J12 - J19)

In addition to all port I/O signals being routed to the 96-pin expansion connector, each of the eight parallel ports of the C8051F040 has its own 10-pin header connector. Each connector provides a pin for the corresponding port pins 0-7, +3.3VDC and digital ground. Table 6 defines the pins for the port connectors. The same pin-out order is used for all of the port connectors.

Pin #	Description
1	Pn.0
2	Pn.1
3	Pn.2
4	Pn.3
5	Pn.4
6	Pn.5
7	Pn.6
8	Pn.7
9	+3 VD (+3.3 VDC)
10	GND (Ground)

Table 6. J12- J19 Port Connector Pin Descriptions

6.8. VDD Monitor Disable Jumper (J23)

The VDD Monitor of the C8051F040 may be disabled by moving the shorting block on J23 from pins 1-2 to pins 2-3, as shown in Figure 4.

MONEN   1
         2
         3
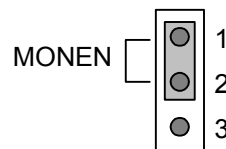
Figure 4. VDD Monitor Hardware Setup

6.9. Expansion I/O Connector (J24)

The 96-pin expansion I/O connector J24 is used to connect daughter boards to the main target board. J24 provides access to many C8051F040 signal pins. Pins for +3 V, digital ground, analog ground and the unregulated power supply (VUNREG) are also available. The VUNREG pin is connected directly to the unregulated +V pin of the P1 power connector. See Table 7 for a complete list of pins available at J24.

The J24 socket connector is manufactured by Hirose Electronic Co. Ltd, part number PCN13-96S-2.54DS, Digi-Key part number H7096-ND. The corresponding plug connector is also manufactured by Hirose Electronic Co. Ltd, part number PCN10-96P-2.54DS, Digi-Key part number H5096-ND.

Pin #	Description	Pin #	Description	Pin #	Description
A-1	+3 VD2 (+3.3 VDC)	B-1	DGND (Digital Gnd)	C-1	XTAL1
A-2	MONEN	B-2	P1.7	C-2	P1.6
A-3	P1.5	B-3	P1.4	C-3	P1.3
A-4	P1.2	B-4	P1.1	C-4	P1.0
A-5	P2.7	B-5	P2.6	C-5	P2.5
A-6	P2.4	B-6	P2.3	C-6	P2.2
A-7	P2.1	B-7	P2.0	C-7	P3.7
A-8	P3.6	B-8	P3.5	C-8	P3.4
A-9	P3.3	B-9	P3.2	C-9	P3.1
A-10	P3.0	B-10	P0.7	C-10	P0.6
A-11	P0.5	B-11	P0.4	C-11	P0.3
A-12	P0.2	B-12	P0.1	C-12	P0.0
A-13	P7.7	B-13	P7.6	C-13	P7.5
A-14	P7.4	B-14	P7.3	C-14	P7.2
A-15	P7.1	B-15	P7.0	C-15	P6.7
A-16	P6.6	B-16	P6.5	C-16	P6.4
A-17	P6.3	B-17	P6.2	C-17	P6.1
A-18	P6.0	B-18	P5.7	C-18	P5.6
A-19	P5.5	B-19	P5.4	C-19	P5.3
A-20	P5.2	B-20	P5.1	C-20	P5.0
A-21	P4.7	B-21	P4.6	C-21	P4.5
A-22	P4.4	B-22	P4.3	C-22	P4.2
A-23	P4.1	B-23	P4.0	C-23	TMS
A-24	TCK	B-24	TDI	C-24	TDO
A-25	/RST	B-25	DGND (Digital Gnd)	C-25	VUNREG
A-26	AGND (Analog Gnd)	B-26	DAC1	C-26	DAC0
A-27	CANRX	B-27	CANTX	C-27	
A-28		B-28	VREF	C-28	VREFD
A-29	VREF0	B-29	VREF2	C-29	HVAIN-
A-30	HVAIN+	B-30	HVCAP	C-30	HVREF
A-31	AIN0.3	B-31	AIN0.2	C-31	AIN0.1
A-32	AIN0.0	B-32	AGND (Analog Gnd)	C-32	AV+ (+3.3 VDC Analog)

Table 7. J24 Pin Descriptions

SILICON LABORATORIES

6.10. VREF Connector (J22)

The VREF jumper block, J22, can be used to connect the VREF (Voltage Reference) output of the C8051F040 to any (or all) of its voltage reference inputs. Install shorting blocks on J22 in the following manner:

> 1-2 to connect VREF to VREFD
> 3-4 to connect VREF to VREF0
> 5-6 to connect VREF to VREF2

7. Serial Adapter

The Serial Adapter provides the interface between the PC's RS232 serial port and the C8051F04x's in-system debug/programming circuitry. The Serial Adapter connects to the C8051F040 JTAG debug interface on the target board using the 10-pin connector on the Serial Adapter labeled "JTAG", see Figure 5. (The Serial Adapter supports both Silicon Laboratories JTAG and C2 debug interfaces.). All Serial Adapters may be powered from the target board, but the EC1 and EC2 Serial Adapter units cannot provide power to the target board. Table 8 shows the pin definitions for the Serial Adapter's JTAG connector.

Notes:
1. When powering the Serial Adapter via the JTAG connector, the input voltage to the JTAG connector's power pin must be 3.0 to 3.6 VDC. Otherwise, the Serial Adapter must be powered directly by connecting the AC/DC adapter to the Serial Adapter's DC power jack.
2. The Serial Adapter requires a target system clock of 32 Khz or greater.

Pin #	Description
1	3.0 to 3.6 VDC Input
2	GND (Ground)
4	TCK (C2DAT)
5	TMS
6	TDO
7	TDI (C2CLK)
3,8,9,10	Not Connected

Table 8. JTAG/DEBUG Connector Pin Descriptions
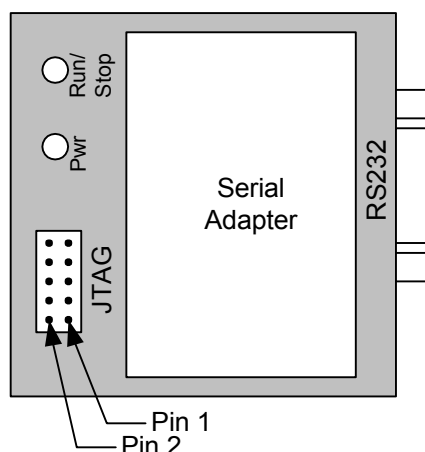
Figure 5. Serial Adapter JTAG/DEBUG Connector

SILICON LABORATORIES
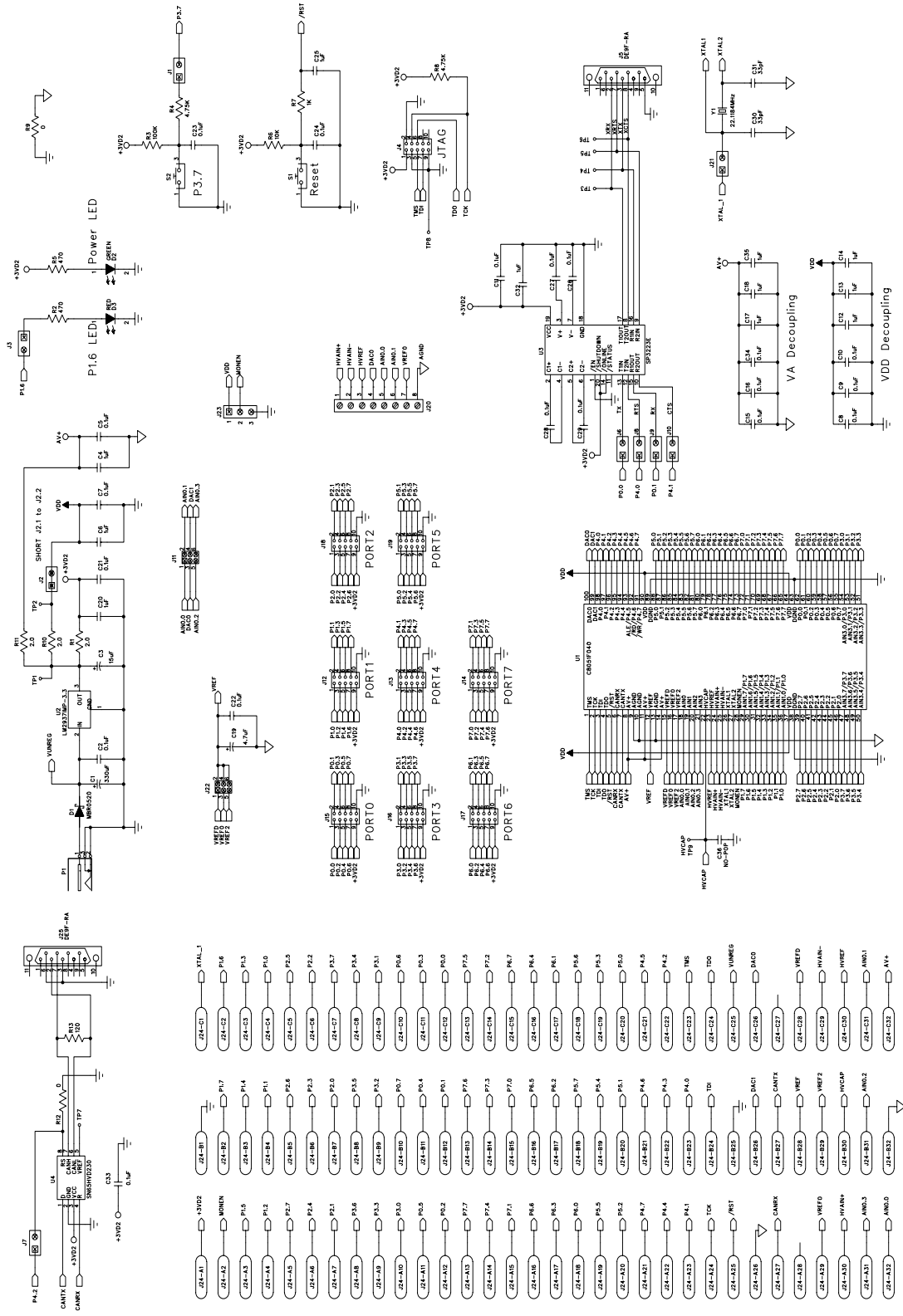
8. Schematic

Figure 6. C8051F040 Target Board Schematic

SILICON LABORATORIES

Notes:

C8051F04x-DK

Contact Information

Silicon Laboratories Inc.
4635 Boston Lane
Austin, TX 78735
Tel: 1+(512) 416-8500
Fax: 1+(512) 416-9669
Toll Free: 1+(877) 444-3032

Email: productinfo@silabs.com
Internet: www.silabs.com

SILICON LABORATORIES