BY DAVID MARSH • CONTRIBUTING TECHNICAL EDITOR

Low-cost ARM kits ease 32-bit migration

WITH 32-BIT MICROCONTROLLERS NOW SO CHEAP THAT THEY RIVAL 8-BIT DEVICES IN VALUE-FOR-MONEY TERMS, A NEW GENERATION OF LOW-COST EVALUATION TOOLS DISPLACES TRADITIONALLY EXPENSIVE DEVELOPMENT SYSTEMS TO EASE ENTRY INTO HIGH-PERFORMANCE EMBEDDED COMPUTING.

For many embedded-system designers, 2005 finally marked the arrival of an array of 32-bit microcontrollers at unprecedentedly low cost. For instance—and as we reported in last December's *EDN Europe*—at $1.47/10,000, Philips set a new low price point for the ever-popular ARM architecture with its LPC2101 (**Reference 1**). This development complemented a host of general-purpose ARM-based microcontroller releases from semiconductor suppliers. At the same time, prices for entry-level devices for proprietary architectures such as Analog Devices' Blackfin fell to the $5 level. But it's not much good having highly affordable chips if you can't afford the development tools, which have traditionally been priced at over $10,000 for many 32-bit environments—and no one spends this kind of money without being absolutely confident that their investment is worthwhile.

As a result, the great majority of semiconductor vendors have woken up to the fact that making products accessible in the first instance is crucially important to secure design wins. All of the companies that we contacted in our December feature stressed the importance of product support, and all offer either their own or third-party evaluation kits to showcase their silicon. Bundling evaluation copies of compilers, debuggers, and integrated-development-environments from software toolmakers such as Green Hills, IAR, Keil, and Tasking gives engineers the opportunity to try these tools before committing to multi-thousand-dollar full-specification compiler purchases. The emergence of on-chip emulation support complements the trend towards lower cost in-circuit emulation, and there are now numerous USB/JTAG devices that permit non-intrusive hardware debugging for orders of magnitude less cost than traditional systems for 32-bit architectures.

Engineers who invest countless hours in becoming familiar with development environments and the architectures that they support need compelling reasons to migrate to new alternatives. Yet one of today's biggest trends is the shift away from 8-bit machines such as generic 8051 derivatives in favour of 32-bit devices that now offer similar microcontroller variety at little extra cost. Confirming this trend, a report in *Electronics Weekly* recently quoted market researchers In-Stat's expectation that by 2009, the combination of 16- and 32-bit revenues will be about twice that of 8-bit devices (**Reference 2**). In-Stat cited the sophisticated features that routinely appear in consumer products as the key driver for migration to wide-datapath cores, noting that higher bit-width microcontrollers require less power than 8-bit devices to do a given amount of work. This is an area where the ARM architecture excels, helping to explain its dominance in portable electronics.

AT A GLANCE

- 32-bit evaluation kits break the $250 marker.
- Targets span analogue microcontrollers to multimedia engines.
- Multiple development-environment choices suit user preferences.
- Low-cost JTAG emulators support non-intrusive debugging.
- Self-training materials complement free code libraries.
- Full-featured environments won't break the bank.

32-BIT CPUs - BEST VALUE?

For engineers working within automotive and industrial spheres, it's typically the raw functionality and deep on-chip memories that 32-bit machines provide that justify making the transition from tried-and-trusted 8-bit devices. Given the low price of 32-bit silicon, it's then germane to question the role of 16-bit machines—especially when devices that employ the ARM7TDMI core bundle a 32-bit CPU with a multiplexed 16-bit external address/datapath that offers cheap memory and peripheral connectivity. Helping to tempt you towards this route, a new generation of affordable entry-level tools endeavours to smooth migration issues into traditionally complex environments. So how approachable are these tools, and how well do they compare with user expectations?

To find out, we invited several key players to submit evaluation kits for ARM-based microcontrollers that suit automotive and industrial applications, with the only limitation being a price ceiling of $500 per kit. The respondents comprise Analog Devices, Atmel, Freescale, and ST Microelectronics, all of whom recently released new silicon that our feature of last December describes. That feature introduces the ARM7TDMI architecture and includes full details for all of the chips in this review. Each microcontroller targets application niches that we'll examine with the help of the resources that come in the box or are available as free web downloads.

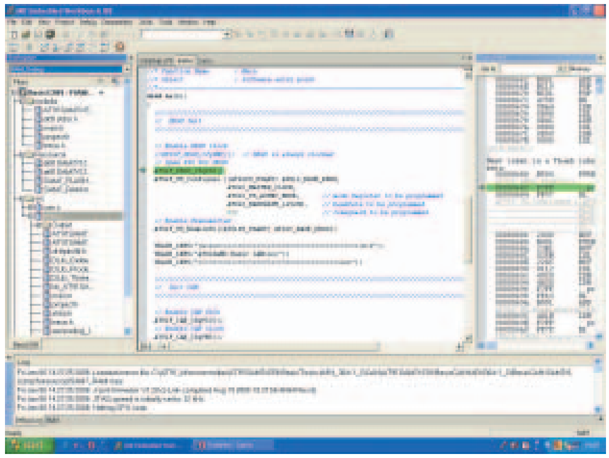

Figure 1 Atmel's SAM7X kit includes numerous training examples, such as this CAN application framework.

The main host PC for these tests is a deliberately typical 2.4 GHz Pentium-4 with 512 Mbytes of RAM, 10/100 Mbps Ethernet, two USB ports, two serial ports, and a parallel port. Separate DVD read and write drives complement dual 80-Gbyte hard disks that hold dual installations of Windows XP Pro SP2 and SuSE Linux Professional 9.3. A second very similar PC running Windows 2000 Pro came into play to avoid corrupting or erasing working examples of similar development environments. It's worth noting that as tested, none of the environments that follow are compatible with Windows 98.

Importantly, each evaluation kit receives two days maximum of exploration time, which should be long enough for the designer to become reasonably familiar with what's on offer. Because the ARM7TDMI core often isn't the best fit for hard real-time applications or for intensive DSP work, an upcoming feature examines proprietary architectures using an identical evaluation kit approach—balancing the generic ARM focus in this article while also presenting alternative solutions for low-power applications. Any of the boards tested in this feature that present compelling reasons to extend the two-day limit will receive a progress update in the later feature. Meantime, see the **sidebar** "Simple Steps Save Frustration" for hints and tips to speed your own evaluation processes.

COMMS CHANNEL BRIDGE

Atmel's latest ARM-series evaluation kit is the AT91SAM7XC-EK that showcases the company's new AT91SAM7X processor. Designed as a communications bridge, this 100-pin chip features CAN, Ethernet, RS232, and USB interfaces, together with hardware support that offloads communication-channel processing overheads from the CPU. The $249 kit contains the evaluation board, an 8-Mbyte Flash card, USB cable, and crossover Ethernet and RS-232 cables. The system documentation and other resources reside on a DVD. There's also a universal-input USB power-supply adapter, complete with US and Continental two-pin plugs. Because the board only accepts power via the USB socket, the USB power supply is helpful when using the JTAG interface for debugging. Be sure to budget $129 for the SAM-ICE in-circuit emulator, which the kit does not include. An Atmel-branded version of Segger's J-Link, the device is next-to-essential, and substantial parts of the DVD's contents require its use.

Available with either 128 or 256 kbytes of Flash, the board carries the processor with the larger on-chip memory. The lack of other chips is testament to the integration that the processor provides—this chip apart, there's a 32-Mbit Atmel serial Flash device, a Philips CAN transceiver, a Davicom Ethernet physical-layer interface, a pair of RS232 interface chips and a rail-to-rail op-amp from Analog Devices, a voltage reference, and a voltage regulator. The interface chips support the board's wealth of I/O ports, including a three-pin analogue interface connector, a CAN port, two nine-pin D-connectors for the RS232 and serial-debug ports, a standard Ethernet socket, and a USB device port. There's also a 20-pin header for the JTAG in-circuit emulator, a slot for the Flash card, and a 0.1in.-pitch, three-row 96-pin expansion connector. A four-axis joystick accompanies the usual complement of LEDs and jumpers.

The "getting started" screen that opens in a web browser offers users the opportunity to explore software-development environments from IAR Systems, Keil, and Green Hills. The role of the joystick soon becomes clear—a general-purpose user-input device, it also simulates a mouse. Having chosen to install Keil's µVision 3, it's also necessary to add an update patch. The next steps install software support for the AT91SAM-ICE in-circuit emulator, without which the kit provides very limited access to the processor. For some unfathomable reason, Atmel chose not to supply this part with the evaluation board, requiring a last-minute search around the UK's distributors to locate one. Thanks to Kim at independent distributor GD-Technik for persuading her warehouse to ship its last example before the seasonal break.

The DVD that accompanies the SAM-ICE contains device-driver software along with a huge amount of information that pertains to the SAM7X's predecessors. Much of this information is still highly relevant and invaluable, but the board's own DVD mirrors the literature version while offering a number of updates. For instance, there's a series of PowerPoint presentations that describe aspects of AT91 operation, from C-code start-up considerations through operation of the family's extensive set of peripherals. There are also example programmes and a mass of application notes, Linux resources, evaluation copies of third-party toolsets, and even board-design files complete with an Orcad-format library. The SAM-ICE device drivers appear in its *tools* subdirectory. When Windows installs these files, the SAM-ICE's LED stops flashing and a success message appears on screen. Click on the *jlink.exe* hyperlink in the browser's set-up window and a DOS window appears to report a successful firmware upgrade. This window echoes firmware revision status, identifies the device by its serial number, and states RDI against the *Features* line—signalling that the hardware complies with the remote-debug-interface specification. This is significant because the SAM-ICE seamlessly integrates RDI support into the target environment, although some restrictions become apparent.

Unzipping a self-extracting project file creates a template to set up the Keil environment. This DVD-resident file creates a new directory, which you then have to point the Keil interface towards—first enabling *see all files* to reveal the crucial *.uV2* extension. The on-screen instructions step through set-up parameters for the SAM-ICE, after which the device is ready to use. With the SAM-ICE and USB power supply connected, it's now time to run the *Getting Started* example project. Four sequentially blinking LEDs signal success. The next exercise burns this programme into Flash, which requires a USB connection to the PC and the installation of Atmel's SAM-BA Flash burner utility. All of these steps completed seamlessly, leaving the system components correctly set up for future use.

Wondering why Atmel promotes its own Flash download mechanisms rather than using the SAM-ICE revealed that an extra-cost Segger licence is necessary for this connection, for the Keil environment at least. Other extra-cost Segger options for this and the J-Link emulators in other kits include the company's J-Link RDI FlashBP software that allows setting multiple breakpoints in Flash. The software costs a substantial €398 within its standard licensing schemes.

During an attempt at familiarisation with the Keil environment and its Flash abilities, the board's USB connection inexplicably failed. Subsequent probing established that the SAM-BA connection to the serial debug port continued to work, as did the SAM-ICE. Suspecting that this fault was due to corrupting the Flash that holds boot code, it was disappointing to find that the *Recovery Procedures* web page heading at www.at91.com—where you will also find a very active user forum—was empty for this board. Jean-Philippe Moreno at Atmel's AT91 support team advises that the correct procedure to recover *SAM-Boot* into the on-chip Flash memory consists of closing the TST jumper (JP5), powering up the kit, and waiting 10 seconds before re-booting.

It also appears that for this kit, the greatest compatibility comes from choosing the IAR toolchain. SAM7X-specific code examples comprise an interrupt handler and a CAN transmitter/receiver routine (**Figure 1**). This is a significant development for Atmel, which in the not-so-distant past has charged as much as $2500 for CAN driver packages to support earlier-generation silicon. There are also two encryption/decryption routines for the XC processor variant that includes scrambler hardware. These routines appear within the DVD's *Software Package* section, where a generic start-up template also exists for each of the software environments. Installing the 32-kbyte-limited Kickstart edition of IAR's Embedded Workbench mirrored the on-screen set-up procedure, which terminates with burning the Flash image via the SAM-ICE without the need for extra-cost licenses. The installation routine builds a 190-Mbyte subdirectory that contains IAR-specific documentation such as user manuals, together with an *AtmelExtras* folder that contains the code examples. In operation, the JTAG debugger routinely autoconnects at an impressive 4 MHz that provides enough responsiveness for the great majority of applications that don't demand hard real-time responses. Any of the routines tested ran faultlessly, complementing the extensive training materials that help to differentiate this kit. Other evaluation-kit designers would also do well to follow Atmel's example in adding full-size ground terminals to allow easy connections to test equipment.

If the 32-kbyte code limit within this or any other IAR Kickstart edition proves insufficient, the tools supplier offers options for all of its ARM-derivative environments. These range from its 256-kbyte-limited baseline version that costs €2200 to its unrestricted €5000 professional edition. The latter version includes 12 months' technical support, network options, and UML documentation facilities. Interestingly, IAR is also working on its own high-performance JTAG debugger that will be available in Q3/2006, with a target price below €1000.

5V SILICON SUITS INDUSTRY

Available now for €260—or approximately $299—the IAR-branded package that contains the STR730-SK/IAR Kickstart Development Board kit is another comprehensively featured environment. It demonstrates STMicroelectronics' latest ARM7TDMI-based family, the STR730 built with the 5V silicon that industrial and automotive designers typically favour. Removing the sleeve reveals two boxes, one of which contains

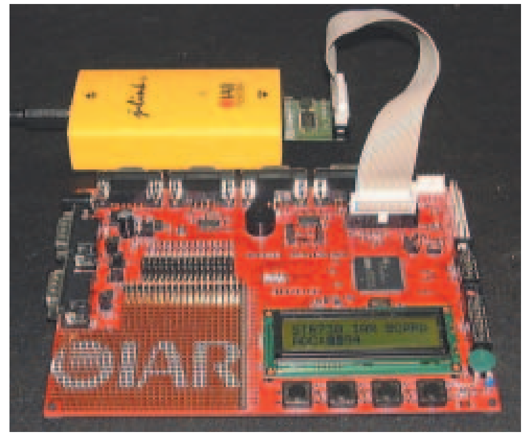

Figure 2 A small adapter connects the 3.3V J-Link emulator to the 5V silicon on IAR's STR730 board.

a J-Link emulator that—colour and branding apart—appears identical to Atmel's SAM-ICE. This version, however, routes 5V to the board from the host PC's USB port via pin 19 of the 20-pin JTAG connector that's unconnected in typical ARM environments. As supplied, the emulator also uses dedicated J-Link rather than RDI drivers. The second box houses IAR's STR730-KS board, a CD-ROM, and an adapter to match the 3.3V J-Link emulator to the microcontroller's 5V levels.

The board seemingly consists of connectors and not much else: apart from the top-specification STR730FZ2 processor that houses 256 kbytes of Flash, 16 kbytes of SRAM, and three CAN interfaces within its 144-pin outline, in chip-count terms there's a pair each of RS232 and CAN transceivers, together with a bridge rectifier and 5V regulator for the optional external power input. Like every kit here, the documentation includes schematics. In this case the circuit prints legibly to one A3 page, attesting to its hardware simplicity. Peripheral devices include a two-line 16-character LCD, four signal pushbuttons and a reset switch, a potentiometer to adjust an analogue-input level, a thermistor, a buzzer, and 16 LEDs. Two CAN ports, four RS232 ports, two I^2C and two SPI ports, as well as headers for the processor's Port-4 and analogue input surround three edges of the board's 180×130 mm outline. There's also a row of pin headers that connect with various I/O lines and a prototyping area (**Figure 2**).

Running the installation routine builds a 196-Mbyte subdirectory that holds

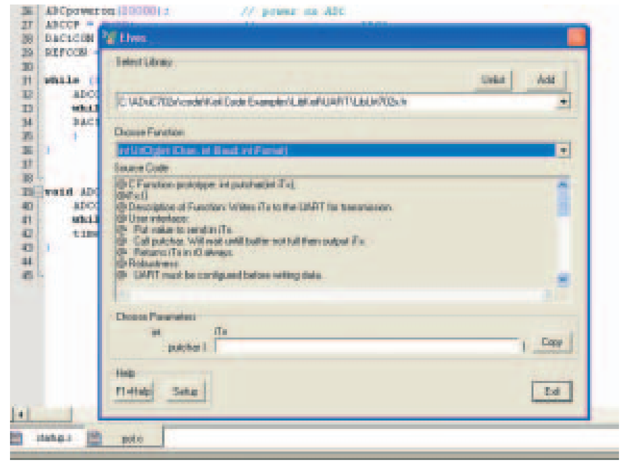

Figure 3 Analog Devices' Elves utility simplifies chip-peripheral configuration tasks.

much the same contents as its Atmel-specific version, including the all-important user manuals. Additional features include a 20-state version of IAR's VisualState application, which includes an automatic code generator with the stealthy ability to construct output from analysing machine states. Target-specific code-example directories for a host of processors appear under the *arm/src/* subdirectory of code examples, including the LCD routine that serves to introduce the system. The lack of a bill-of-material obscures the origin of the LCD module, but it seems to be a standard HD44780-compatible device with the chip-select, *R/W*, and *RS* lines that strobe in data bytes from a four-bit interface port (see http://home.iae.nl/users/pouweha/lcd/lcd.shtml).

While the installation process completed satisfactorily with all drivers, set-up parameters, and switches apparently correctly set, the J-Link emulator refused to communicate with the target board, returning "could not find a device on JTAG chain". Swapping cables and trying the system on another PC that has no previous IAR or J-Link installations returned the same result. The installation didn't include the J-Link manual or the useful-looking *jlink.exe* utility that are available from Segger's website—along with updated drivers that also failed to resurrect the system. 'Scoping out the emulator's JTAG output lines revealed that *TCK* remained high at all times, suggesting a fatal hardware fault. Contacting Product Support revealed (embarrassingly) that your reviewer had installed the adapter board at the target-board end of the cable, not at the emulator end—effectively reversing the adapter. The IDC connection system makes it possible to wrongly connect the components. It's an easy mistake to make as the in-circuit emulator comes with the 20-pin cable attached, making it feel natural to plug other parts into the far end. Our sample is a pre-release version, so expect to find the *jlink.exe* utility included within later versions, along with graphical instructions that even myopic contributing technical editors can't ignore.

With this connectivity problem solved, the board immediately burst into life by running the ADC demo routine that the emulator downloads into Flash. Enabling the *run to main* debugger option and single-stepping from there on in while checking views—such as the processor's registers—effortlessly demonstrates the environment's power. Furthermore, ST's website highlights its new software library that offers a very good-looking support package, including an active user forum. Within these pages appear software libraries for the STR730 as well as for the STR710/720 series, along with all documentation. The software library invaluably contains C-code examples for all the chip's peripherals including the CAN ports, together with start-up projects for ARM's RealView Developer, IAR's Embedded Workbench, and Raisonance's Integrated Development Environment.

MIXED-SIGNAL 32-BIT µCS

While virtually every microcontroller routinely includes ADC support, very few include the analogue blocks that make single-chip analogue processing solutions a reality. A small but rapidly growing part of the microcontroller landscape, mixed-signal processors have typically been 8-bit machines such as the 8051-based products that Analog Devices and Silicon Labs offer. The potential for more sophisticated applications using sensors and other analogue sources led Analog Devices to introduce its ADuC702x family of ARM7TDMI-based microconverters, which its EVAL-ADUC7026QSP kit demonstrates. The $249 box contains the ADuC7026 board, a USB-to-JTAG in-circuit emulator and cables, an RS232 cable with internal level shifters, 120 and 230V two-pin wall adapters, a CD-ROM, and a getting-started booklet.

The two-layer board carries the processor, a 32.786-kHz crystal to generate the phase-lock-loop's 45-MHz clock, 3.3V voltage-regulator and 2.5V voltage-reference chips, a differential-input/output op-amp, a couple of op-amp buffers, and two LEDs to indicate power-good and I/O-pin activity. An 8-way switch offers connection options for the chip's I/O, while three pushbutton switches provide reset, IRQ, and serial download functions. All I/O is accessible from an array of headers. A small demonstration circuit comprising a potentiometer and an LED helps exercise the board's set-up procedure. Pads allow users to add a 16-bit latch to demultiplex address/data for optional 64-kbyte SRAM and 128-kbyte Flash memories. A prototyping area fills the rest of the board's 127×102 mm outline.

The CD-ROM's install routine offers a choice between IAR and Keil software environments with no obvious bias towards either—both offer non-intrusive JTAG debug connectivity via the RDI-compliant Midas-Link emulator. Examining the available software reveals a full GNU toolset for the Keil option, to which the set-up tool and the documentation's commentary defaults. Following this path installs version 2.40c of this environment with additional support for the ADuC7000 series, which offers a comparatively meagre 16-kbyte code limit. It should also silently install the GNU toolchain, but this step required manual intervention. A new directory appears that contains system documentation, code examples, and tools that notably include *Elves.exe*, a configuration utility for the chip's peripherals that isn't automatically installed. The installation process does however add the PLA tool for configuring the programmable-logic-array within ADuC7000-series devices together with a serial-port Flash memory download utility, both of which appear separately under the ADuC702x start-menu entry. The PLA block provides for simple combinational logic control of, say, ADC triggers.

Plugging in the Midas-Link emulator reveals that it too is a Segger device—Windows recognised the device, having previously installed the Atmel-badged version, and automatically assigned the same J-Link driver. The instructions move on to starting a new project, which builds a set-up file complete with parameters for the emulator and the environment for future use. A quick tutorial follows that shows how to add and link a new file to build an application, using as an example one of the chip's DACs to stimulate an ADC channel and return a reading. It's now helpful to examine the Elves facility that holds the promise of simplifying peripheral set-ups. You can run this programme in standalone mode, or integrate it within the IAR or Keil environments by adding it to the Tools menu within these suites. In operation, Elves presents a browser of library functions that you select from a pull-down list, each of which contains a subset of functions. For instance, the UART library includes *getchar* and *putchar* C-function prototypes, among others that set-up and manage the port (**Figure 3**). As well as being simple to use, this system allows users to add their own function libraries providing that they follow a few simple rules that appear in the help menus.

Exploring the *code* subdirectory reveals a wealth of project examples for the GNU, Keil, and IAR environments. Positively the simplest is *blink*, a ten-line routine that alternately turns a LED on and off via a GPIO line. There are separate examples for serial communications comprising I^2C, SPI, and using the UART to complement those that describe the chip's analogue peripherals. Further examples include memory manipulations from placing and running functions from internal/external RAM, through programming and erasing Flash, to placing variables in memory using the GNU compiler. Each code segment is sufficiently simple or has enough inline commenting to be easy to understand, and examples run easily.

SIMPLE STEPS SAVE FRUSTRATION

Testing different and often complex environments suggests several strategies that can ease the process, most of which are plain common-sense but also tempting to ignore. Firstly, consider your objectives–are you simply curious, or are you looking for a solution that will form part of your working environment for years to come? Users looking for upgrade paths from familiar environments will need to feel comfortable with new tools and also be certain of a reasonable degree of future-proofing, as it's both hard work and expensive to make wholesale changes to established practices.

Questions that might apply to any evaluation kit include:

- What's in the box–and what's not–for what cost? Which facilities does the board provide?

- Which development environments are available–and are any of them familiar?

- How good is the supporting documentation–and is any other support available?

- Does the experience persuade me to use the vendor's products?

- If so, how much will it cost to upgrade to a full toolchain?

With any new package, avoid the temptation to plug the board in straight away and rely solely on the quickstart instructions. Sometimes, a quickstart document provides the vital where-to-start link, but it always pays to examine the accompanying documentation and locate which resources are available to guide you–this step alone can save hours of frustration and wasted effort. Don't forget to include the manufacturer's website as part of this process. In particular, look for last-minute software patches and fixes to known problems, as well as software library and application note support. Also, see if there's a user forum and if so, check other users' feedback.

If the kit offers multiple software environments–and unless you're very familiar with one of these–check to see if there's a surfeit of code examples for one in particular, signalling that the board was primarily designed with this environment in mind. When using the environment for the first time, start off with the simplest possible demo routine and take a good look around the environment's controls before writing your own code. At this juncture, it's a good idea to adapt example code if possible. If you're testing complex interfaces such as CAN, make sure that a piece of known working hardware is available to communicate with the board. Starting at around €100, an RS232- or USB-to-CAN adapter from vendors such as Lawicel or Peak can save hours–relying on two brand-new CAN-equipped evaluation boards to talk to each other is a sure recipe for frustration. Other frustrations include power supplies, or the lack of them. Check to see what comes in the bundle, including other essentials such as cables.

Finally, be sure to allocate sufficient time to the project, which for serious users will necessarily be far more flexible than the uniform two-day limit here. While the main text conveys an impression of the relative complexity of the respective environments, it's fair to say that it's far quicker to become fully familiar with a straightforward kit such as the Analog Devices example than with than a complex Linux-powered environment such as Freescale's i.MX. Also, be sure to screen learning time–there's nothing more frustrating than getting the sense that a new aspect is falling into place and then being forced to drop the thread for some unnecessary interruption!

LINUX DRIVES MULTIMEDIA

Designed by Cogent Computer Systems, Freescale's i.MX Litekit just scrapes into our $500 limit with one dollar to spare (suggested retail). It distinguishes itself in several other respects too: targeting multimedia applications, its two boards feature a 200-MHz MC9328MXL processor with an ARM9-derived core; it offers a QVGA graphical touch screen LCD that the processor drives directly, as well as audio capability via a Wolfson codec; it uses Macraigor Systems' usbDemon debugger, an alternative to the JTAG in-circuit emulator interfaces that other products in this article offer; and it comes with a full set of Linux-based tools from Microcross that the company's website prices at $1000. All of these capabilities suit automotive and industrial applications from infotainment to complex man-machine interfaces (see **Figure 4**).

The box contains null-modem RS232, normal and crossover Ethernet, and mini-USB cables, plus a universal-input power supply and a CD-ROM that presents third-party development-support options. The dual-board format consists of a tiny CPU module atop a baseboard that measures about 100×100 mm and carries all I/O, except a mini-USB-format JTAG port that the CPU uses for the Macraigor debugger. Hardware highlights include 64 Mbytes of SDRAM, 8 Mbytes of Flash, slots for Compact Flash and secure digital/multimedia (SD/MMC) cards, the Wolfson stereo codec, and serial, 10 Mbps Ethernet, and mini-format USB device ports. There's also a standard 20-pin JTAG debugger header plus a full-size USB port marked "host" that the design seemingly does not use. The tools install under Windows NT4/2000/XP and Linux for Intel i586/686 architectures—preferably Debian, Linspire, or Red Hat, whose recent versions Microcross has tested for compatibility. Notice that the Macraigor debugger only runs under Windows.

A *readme* file on the i.MX GNU X-Tools CD-ROM describes the software-package contents. This comprises a full GNU compiler toolsuite for the ARM/Thumb processor architecture for OS-less, RTOS, and Linux application development. The constituent parts include ARM/ELF and ARM/Linux compilers, an assembler, debugger, and simulator, the VisualGDB debugger, and libraries. There's also a copy of the Cygwin environment that simulates Linux on Windows hosts—see *EDN Europe*'s recent embedded Linux feature (**Reference 3**). Exploring the docs subdirectory reveals a hardware reference manual for the CSB536FS CPU module and its CSB936FS breakout board, the X-Tools user guide that contains software-installation instructions, and various documents that relate to system components such as the Macraigor debugger. It's noticeable that while the Cogent hardware manual extensively describes I/O connections and gives accurate overviews of the base board's layout, it fails to describe the switches and LEDs on the CPU board—the figure is simply missing, reflecting just how new this kit is.

To install the support environment, Windows users should simply let the Microcross i.MX GNU X-Tools CD-ROM autorun. This process installs a 750-Mbyte *Cygwin* directory structure that transfers all the examples and documentation, including the i.MX GNU X-Tools User Guide. The installation routine creates two desktop shortcuts—*i.MX GNU Tools Shell* and *XWin*, both of which are command-line interfaces. Linux users will first need to refer to the CD-ROM copy of this document for installation instructions. In both cases, sections two and three of this really very good manual continue by explaining how to install the tools and establish communications. The most straightforward connection is 38.4-kbps serial via a terminal programme, such as Windows HyperTerminal. Connect the null modem cable, invoke HyperTerminal, power up the board, and the MicroMonitor startup dialogue appears with its *uMON>* command prompt. This connection accesses the onboard boot monitor program that a 300-page user manual describes. The boot program sets numerous I/O parameters, powers up the LCD, and reports the board's IP address. MicroMonitor promises to save developers massive amounts of time by using its calls to configure and control I/O.

The X-Tools guide shows how to change the network settings to suit the target environment. Because the board's TFTP (trivial-file-transfer-protocol) communications require fixed values, it's advisable to set static server IP addresses if your network runs DHCP (dynamic host configuration protocol). Remove power, plug in the crossover

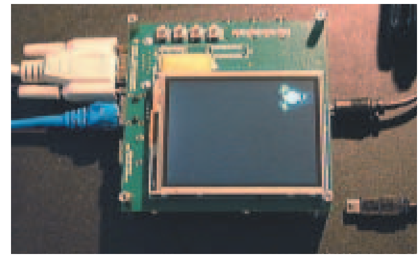

Figure 4 A touchscreen TFT on Freescale's i.MX Litekit enables complex interfaces.

Ethernet cable, re-apply power, and it should be possible to use Windows' *ping* to check the board's Ethernet connectivity at the IP address you set—which MicroMonitor should also report. Alternatively, dispense with the peer-to-peer crossover cable to a PC and plug the board straight into a hub. The user guide steers you through building and loading an application first using MicroMonitor, then GX-Linux. The first process failed, reporting *make: *** No rule to make target 'dld'. Stop.* The reason for this problem remains unclear at press time. By contrast, the complex GX-Linux process worked faultlessly, building the environment and uploading it to the board's Flash. Typing *startlinux* invokes the environment and readies it for running the test programs.

To configure the usbDemon debugger, you now install OCD Commander, a piece of Macraigor freeware that's deposited under the Cygwin tree. Installing this software adds another two desktop shortcuts, together with two drivers that enable usbDemon to communicate—which the usbDemon Finder utility proves by flashing a LED on the CPU board and returning a device serial number that's needed to invoke OcdRemote. This standalone utility runs under the X-Tools shell, listening to a TCP/IP port and converting incoming GDB-debugger commands to JTAG signals. To get this to work, the documentation describes how to make a test program within the X-Tools shell prior to connecting the null-modem and Ethernet cables. Invoke another instance of X-Tools and start OcdRemote with a command line such as *ocdremote –c ARM920T –d USB –a 0 –s 1*, when a cryptic message appears that confirms that the utility is listening on its default port 8888.

Windows XP Pro also reports a 10-Mbps connection to the board's low-power Ethernet port at this point. It's nec-essary to repeat this connection process every time that you start a new debugger session.

With all communications established, it's now possible to run the command-line GDB debugger or its graphical equivalent, VisualGDB. Typing *arm-elf-gdbtk test.x* into the first X-Tools instance selects the graphical option. It presents a familiar-looking windowed environment, with typical commands such as set breakpoints and single-step complementing views such as registers and memory (**Figure 5**). This proves that the environment is set up and working, and it will remain set until the user makes any changes. The console window provides additional monitor commands to those available directly from the graphical source window. The documentation continues to describe how to install and run a 30-day evaluation copy of Visual X-Tools, a $500 Microcross product that seamlessly combines the GNU tools and debuggers within an integrated-development-environment an option that's likely to appeal to programming professionals.

From this point onwards, anyone who has worked with Linux before will sense a familiar environment. Because it provides so many basic resources as well as useful programming examples to control the board's I/O, it's essential to master the MicroMonitor system as well as exploring the Linux environment. How long the familiarisation process will take depends very much on individual experience, but your reviewer simply ran out of time at this point. Watch out for an update later on this year within our proprietary architectures coverage. **EDN**

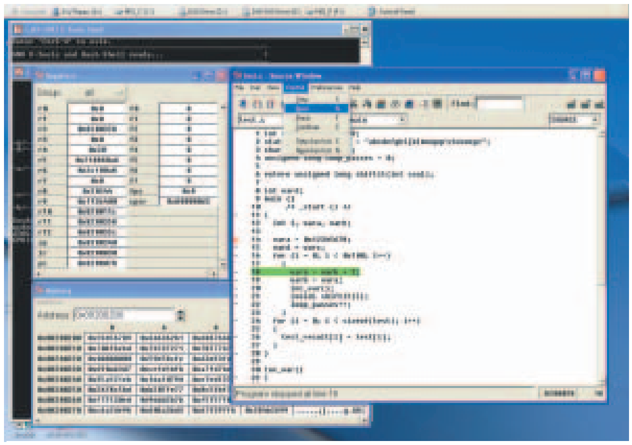

Figure 5 Macraigor's usbDemon and VisualGDB debug Linux applications.

REFERENCES

1. Marsh, David, "ARM targets automotive and industrial dominance," *EDN Europe*, December 2005, pg 24, www.edn-europe.com.
2. "War of the worlds: 16- and 32-bit MCUs invade ASIC applications," In-Stat 2005, Report price: $3,995, www.in-stat.com.
3. Marsh, David, "Embedded Linux steals design wins," *EDN Europe*, June 2005, pg 20, www.edn-europe.com.

AUTHOR'S CONTACT INFO

You can reach Contributing Technical Editor David Marsh at forncett@btinternet.com.

FOR MORE INFORMATION

Analog Devices
www.analog.com

ARM
www.arm.com

Atmel
www.atmel.com

Davicom Semiconductor
www.davicom.com.tw

Freescale Semiconductor
www.freescale.com

GD-Technik
www.gd-technik.com

Green Hills Software
www.ghs.com

IAR Systems
www.iar.com

Keil Software
www.keil.com

Lawicel
www.lawicel.com

Macraigor Systems
www.macraigor.com

Microcross
www.microcross.com

Peak-System Technik
www.peak-system.com

Philips Semiconductors
www.semiconductors.philips.com

Raisonance
www.raisonance.com

Segger Microcontroller Systems
www.segger.com

Silicon Labs
www.silabs.com

SofTec Microsystems
www.softecmicro.com

STMicroelectronics
www.st.com

Tasking
www.altium.com

Wolfson Microelectronics
www.wolfsonmicro.com