

Probe Configuration for Dev Board



Application Note 3 Sensor Node Software

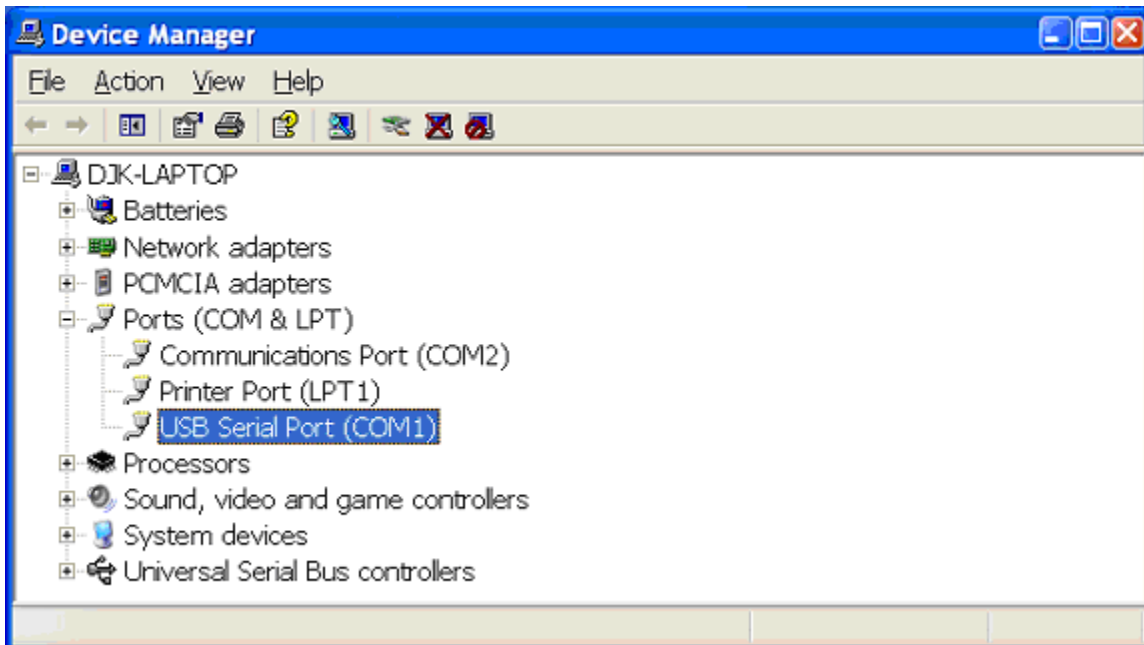
Configuring a Green Hills Probe for Use with the GainSpan Development Board

INTRODUCTION

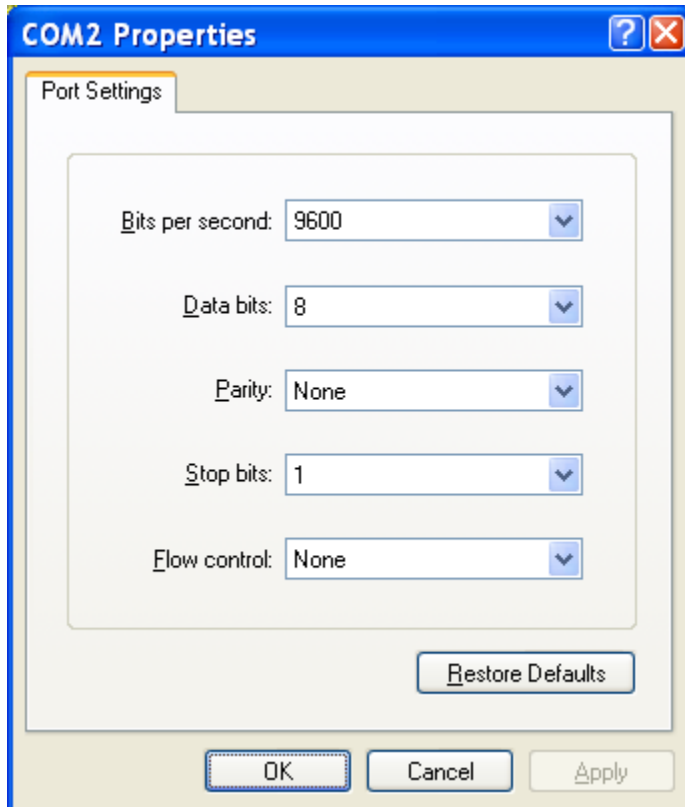
WHAT FOLLOWS is a step-by-step guide to configuring a Green Hills Probe (Probe) for use with a GainSpan GS1010 Development Board (DB).

CONFIGURATION DIRECTIONS

1. Start with Probe and DB both powered off.
2. Attach Probe to PC using RS232 cable to a free COM port (e.g., COM1). *NOTE: In some cases, such as using a USB-to-RS232 cable, it may not be obvious to which COM port an RS232 cable is attached. This is can be found by examining the Device Manager: Start → Control Panel → System → Hardware → Device Manager → Ports (COM & LPT).*



3. Attach the Probe to the LAN using an Ethernet cable. Ideally a DHCP server is accessible on the LAN. *NOTE: It is also possible to configure the Probe to use a static IP Address/Subnet/Gateway, but the steps are slightly different. This manual assumes DHCP is being used.*
4. Attach the 20-pin ARM ribbon cable between the Probe and the DB JTAG port (J1).
5. Assure that the DB is in BootROM mode with JTAG of both WCPU and ACPU enabled. In other words, set the following DB switches (INT2):
 - A. S3 = ON (S3 = On: Stay in BootROM)
 - B. S4 = OFF
 - C. S5 = ON ({S4,S5} = {Off,On}: “Enable WCPU and ACPU JTAG”)
6. Power up the DB. *NOTE: There are many power configuration options for the DB. This manual assumes that a method for powering the DB is already established.*
7. On the PC, open a terminal program (e.g., *HyperTerminal*, from Start → Accessories → Communications → HyperTerminal).
8. Open a new connection using the COM port from step 2 and the following settings:



9. Power on the Probe – turn on the switch from O → I. The Probe will send boot up spew to the terminal program (Tera Term is shown).

```

COM1 - Tera Term VT
File Edit Setup Control Window Help
ghStarting.
Reading Configuration from non-volatile memory.

Green Hills Probe and SuperTrace Probe for ARM/XScale, v1.12.2
Firmware built Sep 5 2006 12:16:09
Serial No: 8958
Part number 520-GP300-02A
Green Hills Software, Inc
http://www.ghs.com

Serial:      9600
USB:        Enabled
Ethernet:    Enabled
IP:         DHCP
Gateway:    DHCP
Netmask:    DHCP
MAC Addr:   00:12:5c:00:1d:fe

Starting the initialization sequence.
01234567 done.

Type 'info' to list probe information.
Type 'setup' to set the current configuration.
Type 'help' to list the online help.

GreenHills Probe
DHCP IP address bound: 192.168.3.106
Gateway bound: 192.168.3.2
arm7tdmi-s[0,?] % █

```

10. If the DHCP was already configured on the Probe and the LAN, the Probe's IP address will be displayed as highlighted in the Tera Term screenshot. Note the IP address and skip to step 12.
11. If the DHCP was not already configured, setup the Probe as follows:

```

arm7tdmi-s[0,?] % setup
Green Hills Probe (tm) Interactive Setup

To accept the current setting press 'Enter'.
For information on any option type a '?'.
Use the 'set' command at a terminal to individually set options.

Network configuration (net):

dhcp: [off] on
NOTE: You have to reboot for this option to take effect.
dhcp_lease: [0 days] 0

Target Interface configuration (trg):

adapter: [arm-20] arm-20
logic_high: [3.300000] 3.3
target: [arm7tdmi-s arm7tdmi-s] arm7tdmi-s arm7tdmi-s

Target configuration (core0):

endianness: [little] little
disable_swbp: [off] off

Target configuration (core1):

endianness: [little] little
disable_swbp: [off] off

Saving options to EEPROM...

```

```
Options have been changed that require the probe to be rebooted before
taking effect:
```

```
Reboot now [yes] yes
```

```
Rebooting probe...
```

```
ghStarting.
```

```
Reading Configuration from non-volatile memory.
```

```
Green Hills Probe and SuperTrace Probe for ARM/XScale, v1.12.2
```

```
Firmware built Sep 5 2006 12:16:09
```

```
Serial No: 8958
```

```
Part number 520-GP300-02A
```

```
Green Hills Software, Inc
```

```
http://www.ghs.com
```

```
Serial:      9600
USB:         Enabled
Ethernet:    Enabled
IP:          DHCP
Gateway:     DHCP
Netmask:     DHCP
MAC Addr:    00:12:5c:00:1d:fe
```

```
Starting the initialization sequence.
```

```
01234567 done.
```

```
Type 'info' to list probe information.
```

```
Type 'setup' to set the current configuration.
```

```
Type 'help' to list the online help.
```

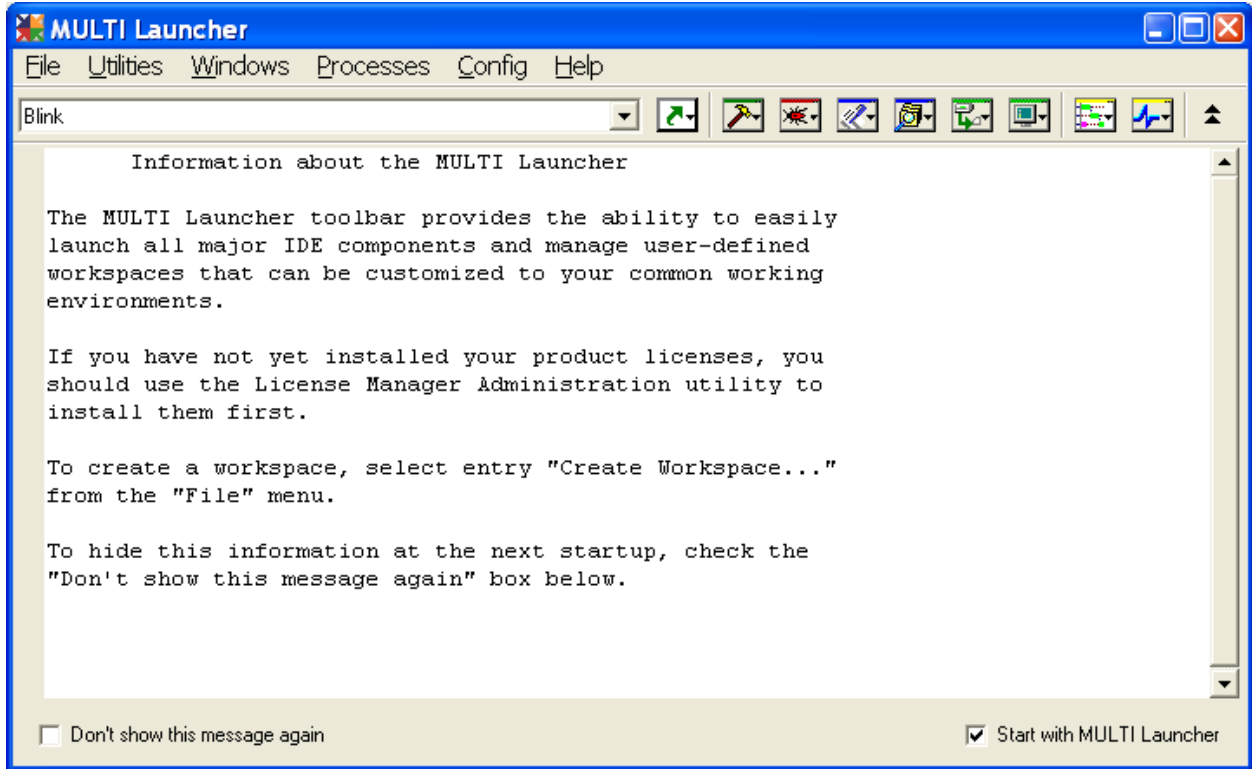
```
GreenHills Probe
```

```
DHCP IP address bound: 192.168.3.106
```

```
Gateway bound: 192.168.3.2
```

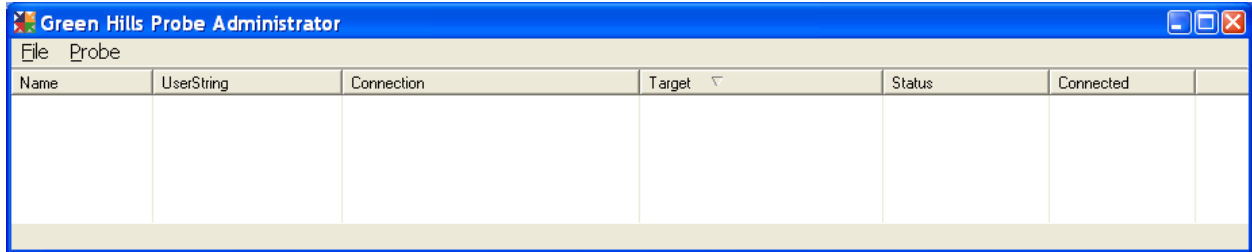
```
arm7tdmi-s[0,?] %
```

12. If the DHCP and the LAN are configured properly, the new DHCP-assigned IP address is displayed when the Probe reboots.
13. Start *MULTI*. This opens MULTI Launcher.

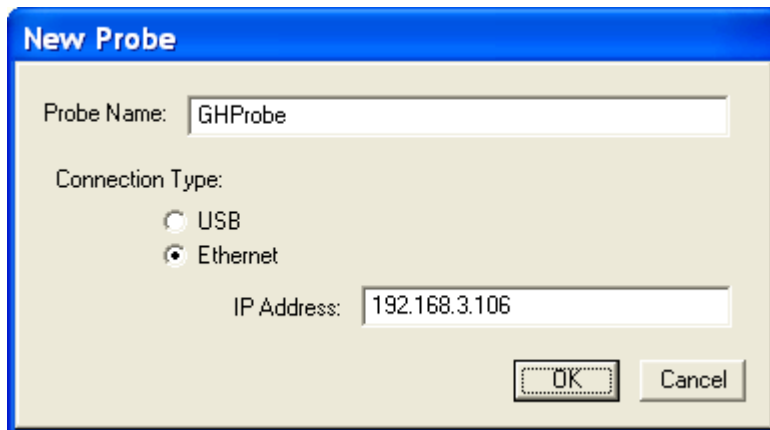


14. Select menu option Utilities → Probe Administrator.

15. This opens the Green Hills Probe Administrator.



16. Create a new Probe (File → New Probe...) using the Probe IP Address assigned by the DHCP.



17. The new Probe is now displayed in the Green Hills Probe Administrator.

Name	UserString	Connection	Target	Status	Connected
GHPProbe	GreenHills Probe	Ethernet [192.168.3.106]	arm7tdmi-s arm7tdmi-s	target power of	Disconnected

18. Right click on the newly added probe and select Configure Probe...

Probe Administration - GHPProbe (GreenHills Probe)

File

Info Configuration Prompt

Option	Value
<input type="checkbox"/> System	
dhcp	on
user_string	GreenHills Prok
baudrate	9600
<input type="checkbox"/> Target Interface	
adapter	arm-20
clock	1 MHz
logic_high	3.300000
checker	on
power_detect	on
rst_pulse	300 ms
rst_settle	300 ms
jrst_pulse	300 ms
jrst_settle	300 ms
<input type="checkbox"/> Other	
dhcp_ip	192.168.3.106
dhcp_netmask	255.255.255.0
dhcp_gateway	192.168.3.2
dhcp_lease	0 days
dhcp_hostname	ghprobe8958
single_mpserv_on	off
rtck_use_timeout	on
user_button	tri-state
rst_tap_then_sys	off
<input type="checkbox"/> Target	
use_rtck	off
target	arm7tdmi-s arm7tdmi-s
<input type="checkbox"/> Core 0: arm7tdmi-s	
endianness	little
fast_dl	off
abort_check	on
disable_swp	off
rst_type	hard
override_mmu	off
force_dbgack	on
halt_settle	300 ms
<input type="checkbox"/> Core 1: arm7tdmi-s	
endianness	little
fast_dl	off
abort_check	on
disable_swp	off
rst_type	hard
override_mmu	off
force_dbgack	on
halt_settle	300 ms

target

Format:

```
set target $target
```

Describes the target and/or scan chain the probe is attached to. The list of all devices in the chain, the first device in the list is the first. Devices that the probe does not support can be described as "other" device. For a single-core configuration, only a single device is specified.

Examples:

```
set target arm7tdmi # Tells the probe it's connected to an ARM core.
set target arm7tdmi mips32_4ksc # Tells the probe it's connected to a MIPS 4KSc.
set target arm7tdmi other(4,0xf,1) mips32_4ksc # Tells the probe it's connected to a regular JTAG device that is some unsupported device (Ethernet PHY) with an instruction that is 4 bits wide, a bypass data length of 0xf, and a bypass data length of 1 bit wide. The third core is a MIPS 4KSc core.
```

Legal Values:

a list of JTAG scan chain devices.

Supported targets:

Generic JTAG devices:

```
other(<instruction length>,<bypass instruction>,<bypass data length>)
```

ARM Targets:

ARM7 Targets:

```
arm720, arm7di, arm7tdmi, arm7tdmi-s, mac7111
```

ARM9 Targets:

```
arm920, arm922, arm925, arm926, arm940, arm946, arm966, arm968
```

ARM10 Targets:

```
arm1020, arm1022
```

ARM11 Targets:

```
arm1136
```

set target arm7tdmi-s arm7tdmi-s Apply

Core 0 halted pc=0xe001017f

19. Activate the Configuration tab, and set the properties per the screenshot (except for the DHCP settings, which should not be modified). Some important parameters to verify are:

Parameter	Value	Description
clock	1MHz	JTAG clock: This should almost always be set to 1MHz. The only exception is for the extremely rare case when debugging code in DeepSleep or the Slow states, described in the JTAG Requirements section of this document. In these cases, set the clock to 10 kHz.
logic_high	3.300000	JTAG signal voltage: When the DB JTAG I/O voltage is configured for 3.3V (the default configuration), this should be set to 3.300000. The DB JTAG I/Os are all on I/O bank 1, so its voltage is set by vddio_bank1 (A8), which on the DB is selected using J9. When the J9 jumper is between pins 1-2, 3.3V is selected; between 2-3, it is 1.8V. In both cases, jumpers J7 and J15 should both be installed. If 1.8V is selected for bank 1, set logic_high = 1.800000.
target	arm7tdmi-s arm7tdmi-s	JTAG can be used to debug both CPUs (almost) simultaneously. This setting reflects that configuration. If only one CPU has JTAG enabled, set target to arm7tdmi-s. Enabling CPU JTAG is described in the JTAG Requirements section of this document.
endianness	Little	The CPUs are both always Little-Endian.
fast_dl (both Cores)	Off	JTAG can download compiled code both to RAM and to the flash. For flash programming, however, the <i>MULTI</i> fast_dl must be disabled. With this option enabled, <i>MULTI</i> writes memory as fast as it can. This is too fast for the flash when using a 1MHz JTAG clock. This could be set to on when performing RAM debugging, however, for a 300% write speed improvement.
halt_settle (both Cores)	300msec	How long <i>MULTI</i> waits to determine whether a JTAG target halt (th) command succeeded.

JTAG REQUIREMENTS

The GS1010 System-On-Chip (SOC) contains two ARM7 (ARM7TDMI-S to be precise) CPUs. One is the master CPU (WCPU), dedicated to WLAN MAC processing and system bootup. The other is the Application processor (ACPU), which runs the *GHS uVelOSity* RTOS, the *GHS GHNet* network stack, *GainSpan Node Software*, and all the sensor applications. Both processors can be debugged using JTAG.

However, four things are required for a CPU's JTAG to function properly:

1. Power (core voltage = 1.8V)
2. JTAG enabled
3. Reset released
4. Clock

Power

Both CPU cores are powered from the same 1.8V regulator. The Real Time Counter (RTC) is powered directly from a battery and controls this regulator using the DC_DC_ENABLE pin. Thus, the system can save power by having the RTC disable the regulator, completely powering down the CPUs (as well as RAM and the vast majority of the SOC). When the RTC wants to reawaken the system, it re-enables the regulator. The state when the RTC is powered, but the rest of the chip is not, is called Standby. JTAG does not work for either CPU when the SOC is in Standby.

JTAG Enable

The two CPUs can have their JTAG debug ports independently enabled or disabled using DB switches (S4 and S5). These switches are sensed only on Power On Reset (POR) or External Reset (EXT_RST).

SW4	SW5	JTAG Enabled?	
		ACPU	WCPU
Off	Off	Yes	No
On	Off	No	Yes
Off	On	Yes	Yes

Reset & Clock

From a hardware perspective, the JTAG block is an integral part of the CPU core. Thus, it shares both the CPU's clock and reset. So, when a CPU is held in reset, or its clock is gated (which means that it is not clocked), the JTAG does not function. The SOC has various Sleep modes where one or the other CPU is either held in reset or has its clock gated to save power.

The SOC has special hardware that detects JTAG accesses during these modes and either de-asserts reset or un-gates a clock as required for the duration of the JTAG access. In this way, JTAG is enabled even in these sleep modes. How exactly this hardware works depends on which CPUs have their JTAG enabled:

JTAG configuration	Description
ACPU only	On the first JTAG access, the ACPUs are automatically released from reset, and enabled (APPRESET and APPENABLE register fields are set). On every JTAG access, the ACPUs clock is un-gated.
WCPU only	On every JTAG access, the WCPUs clock is un-gated.
ACPU & WCPU	On the first JTAG access, the ACPUs are automatically enabled (the APPENABLE register field is set). <i>NOTE: APPRESET, however, is still controlled by the WCPU.</i> On every JTAG access, both the WCPUs and ACPUs clocks are un-gated.

The SOC uses two clock sources:

1. A 32kHz low-power clock (LPC), which is used primarily for clocking the Real Time Clock (RTC) block.
2. A 44MHz clock used to generate the radio frequencies and also, usually, to clock the CPUs.

Generating the 44MHz clock consumes a significant amount of power, and therefore, to optimize power savings, this clock is often shut off. When this happens, the CPUs can still be clocked by the LPC. This gives the “slow” run modes.

This is important to note, since the internal JTAG block shares the CPU clock. This clock must always be faster than the externally applied JTAG clock. Thus, when using JTAG during a “slow” run mode (i.e., CPU clocked at 32kHz), the external JTAG clock must be significantly less than 32kHz. (For the Probe, the minimum frequency is 10kHz.)

Run Mode Summary

The GS1010 includes many run modes for optimal power consumption. Whether or not JTAG debugging will work is heavily dependent on in which run mode it is operating. The following table summarizes these modes, and the JTAG clock recommended for each mode. *NOTE: in some modes JTAG will not work at all (JTAG Clock = '-').*

Run Mode	WCPU				ACPU			
	Exec State	Clock			Exec State	Clock		
		CPU	State	JTAG		Freq	State	JTAG
Off	Off	-	-	-	Off	-	-	-
Standby	Off	-	-	-	Off	-	-	-
SysReset	Reset	-	-	-	Reset	-	-	-
SysConfig	Run	44MHz	Enabled	1MHz	Reset	44MHz	Gated	1MHz
Fast	Run	44MHz	Enabled	1MHz	Run	44MHz	Enabled	1MHz
FastACPU	Sleep	44MHz	Gated	1MHz	Run	44MHz	Enabled	1MHz
FastWCPU	Run	44MHz	Enabled	1MHz	Sleep	44MHz	Gated	1MHz
Sleep	Sleep	44MHz	Gated	1MHz	Sleep	44MHz	Gated	1MHz
Slow	Run	32kHz	Enabled	10kHz	Run	32kHz	Enabled	10kHz
SlowACPU	Sleep	32kHz	Gated	10kHz	Run	32kHz	Enabled	10kHz
SlowWCPU	Run	32kHz	Enabled	10kHz	Sleep	32kHz	Gated	10kHz
SysConfigSlow	Run	32kHz	Enabled	10kHz	Reset	32kHz	Gated	10kHz
DeepSleep	Sleep	32kHz	Gated	10kHz	Sleep	32kHz	Gated	10kHz

Each processor can be in one of four execution states (Exec State):

Exec State	Description
Off	No power supplied to core (1.8V regulator is off). Both CPUs are supplied from same V_{dd} , so both are always off at the same time.
Reset	Each CPU has an internal RESET signal. The WCPU has four reset sources: WDOG, SW, POR, EXT_RESET. The ACPU has five reset sources: WDOG, SW, POR, EXT_RESET, WCPU. <i>NOTE: The WCPU is the master CPU. This is a very critical point for JTAG debugging. After power is applied, the ACPU stays in reset until released by the WCPU. ACPU JTAG will not work until this has occurred.</i>
Run	CPU is being clocked, meaning it is fetching and executing instructions.
Sleep	CPU clock is gated, i.e., CPU is not clocked. There is special hardware, however, that will clock the CPU when JTAG signals are detected.

TESTING THE JTAG CONNECTION

To test the JTAG connection, please use the *Blinky* sample project. This patent-pending piece of code blinks two LEDs on the DB. Getting it to work, however, takes some effort.

1. Start with the Probe and the DB both powered off.
2. Assure that the DB is in BootROM mode, with JTAG of both WCPU and ACPU enabled. In other words set the following DB switches (INT2):
 - A. S3 = ON (S3 = On: Stay in BootROM)
 - B. S4 = OFF
 - C. S5 = ON ($\{S4,S5\} = \{Off,On\}$: "Enable WCPU and ACPU JTAG")
3. Power up the DB. The four LEDs in a row (D1, D2, D6, D7) should be (off, on, off, off).

4. Power up the Probe.
5. On the PC, open a terminal program. Open a telnet connection to the Probe IP address.
The Probe sends telnet welcome spew, ending in an `arm7tdmi-s[0,f] %` prompt. This prompt indicates that target is Core 0, which is an `arm7tdmi-s`, and it is free-running. Core 0 is the WCPU when both CPUs have JTAG enabled. Free-running means that the Probe is not really connected to its JTAG.
6. Reset WCPU by entering the command `tr`.
This actually resets both cores (Core 0 and Core 1 = WCPU and ACPU). The Probe reports the CPU's PC register value when the cores halt. The PC is the "Program Counter" and stores the location of the next instruction to be executed by the core.
7. All the other WCPU ARM7 registers can be read by the command `rr`.

```

192.168.3.106 - Tera Term VT
File Edit Setup Control Window Help

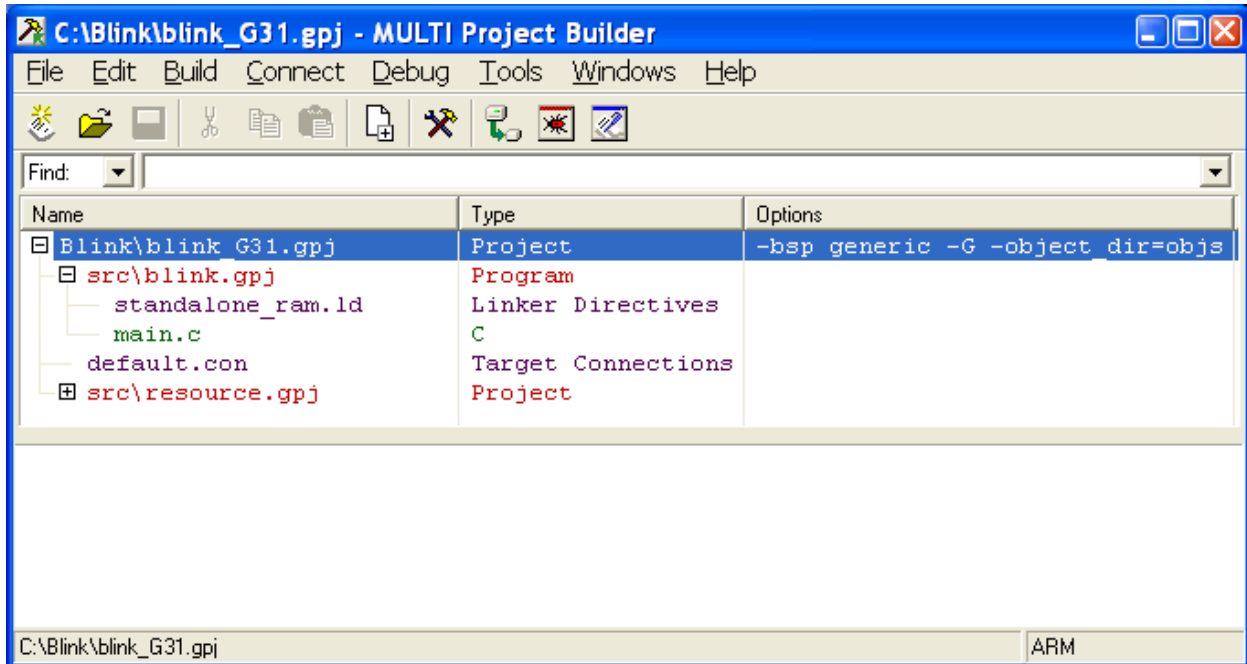
Green Hills Probe and SuperTrace Probe for ARM/XScale, v1.12.2
Firmware built Sep  5 2006 12:16:09
Serial No: 8958
Part number 520-GP300-02A
Green Hills Software, Inc
http://www.ghs.com

Type 'info' to list probe information.
Type 'setup' to set the current configuration.
Type 'help' to list the online help.

GreenHills Probe
arm7tdmi-s[0,f] % tr
arm7tdmi-s[0,h] %
Core 1 halted; pc=0x0025a2f8
arm7tdmi-s[0,h] %
Core 0 halted; pc=0x02000810
arm7tdmi-s[0,h] % rr
r0=0x00000000      r9=0x00000000      r1_usr=0x05000000      r10_usr=0x00000000
r1=0x05000000      r10=0x00000000      r2_usr=0x0100000c      r11_usr=0x00000000
r2=0x0100000c      r11=0x00000000      r3_usr=0x02002754      r12_usr=0x020029a0
r3=0x02002754      r12=0x020029a0      r4_usr=0x05000000      sp_svc=0x01002e54
r4=0x05000000      sp=0x01002e54      r5_usr=0x00000000      lr_svc=0x02000818
r5=0x00000000      lr=0x02000818      r6_usr=0x00000000      spsr_svc=0x00000010
r6=0x00000000      pc=0x02000810      r7_usr=0x00000000
r7=0x00000000      cpsr=0x60000053      r8_usr=0x00000000
r8=0x00000000      r0_usr=0x00000000      r9_usr=0x00000000
arm7tdmi-s[0,h] % █

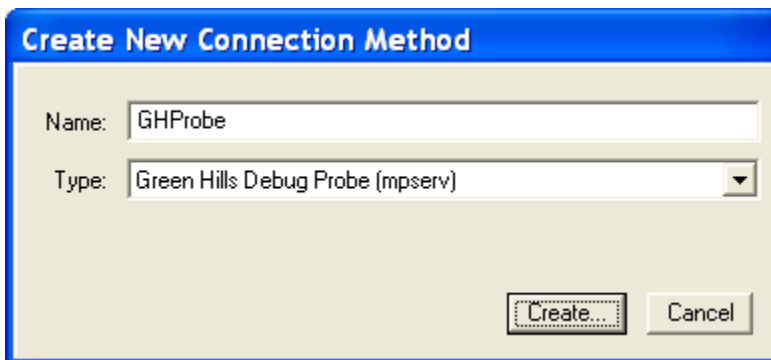
```

8. Start *MULTI*. This opens MULTI Launcher.
9. Open the `Blink\blink_G31.gpj` GHS project: File → Open Project Builder...

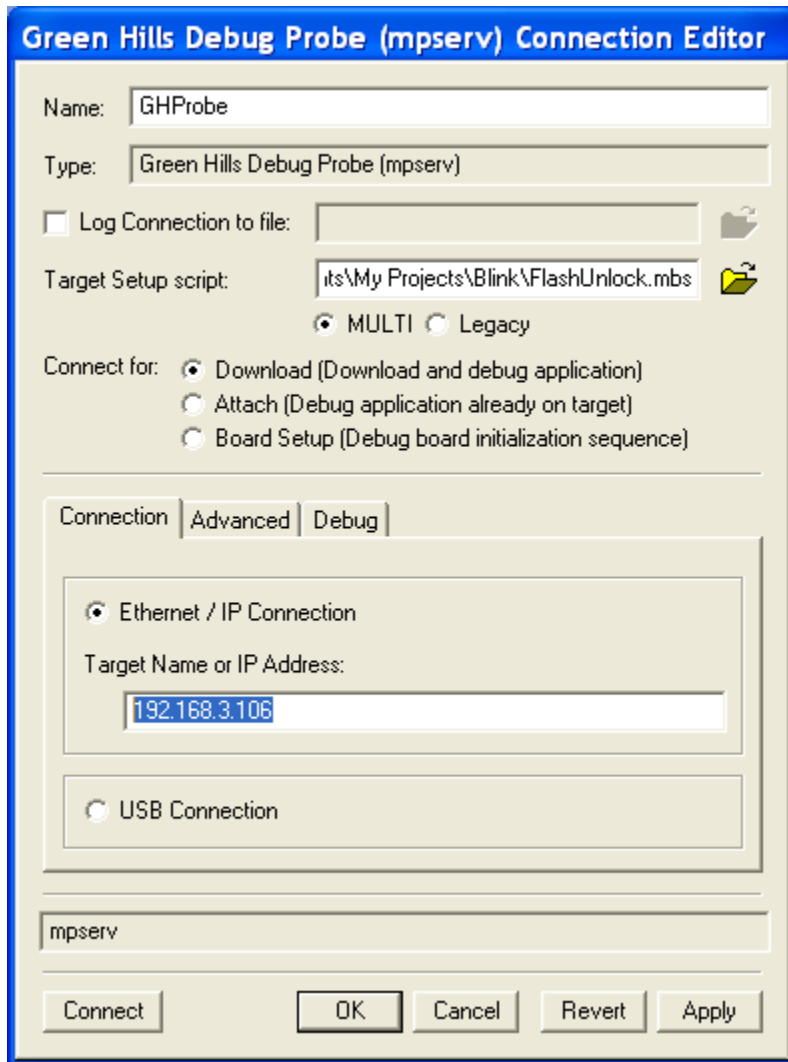


This project contains a single source file (`main.c`) and a linker script (`standalone_ram.ld`) that will place the compiled code in RAM. Debugging code running from RAM (as opposed to from the flash) allows setting multiple breakpoints. Flash debugging is restricted to two hardware breakpoints, one of which is reserved for single stepping code.

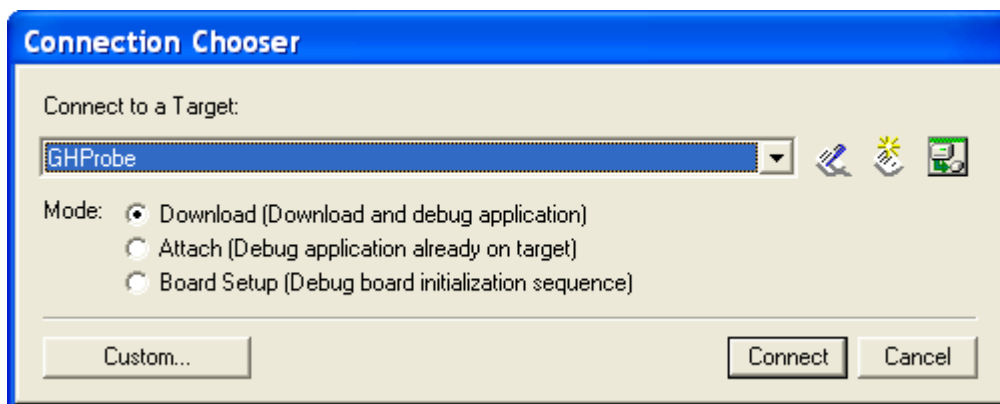
10. Create a new connection using the Connection Organizer: **Connect** → **Connection Organizer**.
11. Select **Method** → **New...**



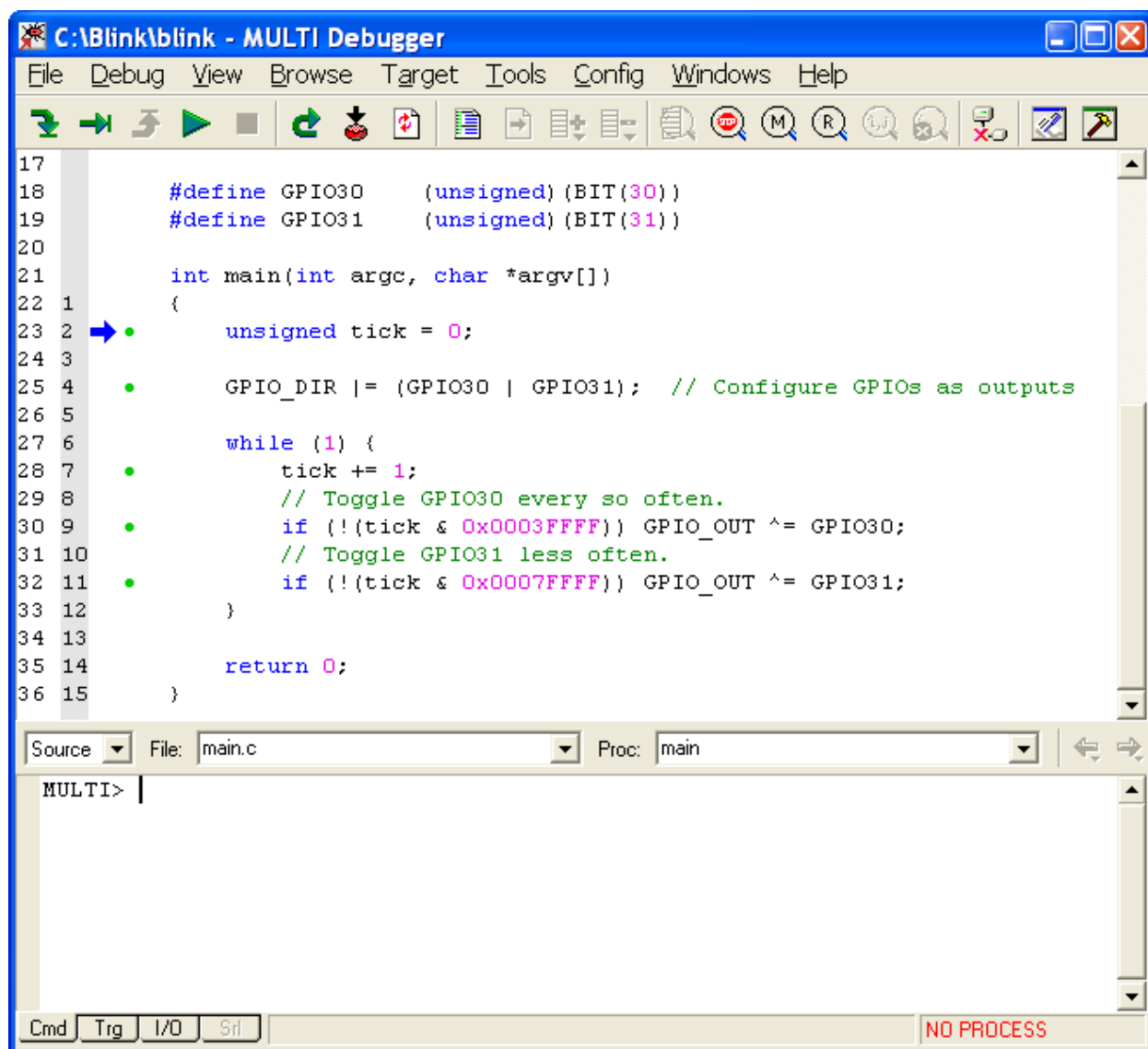
12. For **Type**, select **Green Hills Debug Probe (mpserv)**, and name it something like **GHProbe**. Then click **Create...** to open the **Green Hills Debug Probe (mpserv) Connection Editor**.



13. Click the open file button and select Blink\FlashUnlock.mbs as the Target Setup script.
This script will use JTAG to write “magic values” to a flash controller register which will unlock the two APP flashes. Although this project only writes to RAM, not to the flash, this same Target Setup script will be used later to write to the flash.
14. Select Ethernet / IP Connection and enter the Probe IP Address, as highlighted in the screenshot.
15. Select the Advanced tab.
16. Select Force core ID# checkbox, and enter 1 in the box.
This will force *MULTI* to use Core 1 which corresponds to the ACPU when JTAG is enabled on both ACPU & WCPU (S4 = off, S5 = on).
17. Click OK.
18. Close the Connection Organizer.
19. From the MULTI Project Builder, connect to the Probe using Connect → Connect...



20. Select GHProbe and Download (Download and debug application), and then click Connect.
21. From the MULTI Project Builder, select the Project file (Blink\blink_G31.gpj) and debug it using Debug → Debug blink.



22. Run the program: Debug → Go.
 23. The LEDs do not blink!!! Why???
- The WCPU is the master CPU. On reset, the WCPU is in control of all of the shared peripherals (ADC, UART, I²C, SPI), including all of the GPIOs. Only the WCPU can change the ownership of these peripherals or GPIOs. Unfortunately, we are debugging code running on the ACPU (Core 1), so this code cannot directly take control of the GPIO. For that, we must talk to the WCPU directly.
24. Halt the program: Debug → Halt.
 25. Return to the telnet window.
 26. Observe the Core 1 messages that were generated when running the program on ACPU in the debugger.
 27. Read the contents of the SHGPIO register: mr 0x0500010C.
0x0500010c: 0x00000000
All GPIOs are currently assigned to WCPU.
 28. Assign GPIO30 and GPIO31 to ACPU: mw 0x0500010c 0xC0000000.
 29. Restart Blink in the debugger Debug → Restart.

Congratulations... D6 and D7 should now be blinking!!!

```

192.168.3.106 - Tera Term VT
File Edit Setup Control Window Help
Type 'setup' to set the current configuration.
Type 'help' to list the online help.

GreenHills Probe
arm7tdmi-s[0,f] % tr
arm7tdmi-s[0,h] %
Core 1 halted; pc=0x0025a2f8
arm7tdmi-s[0,h] %
Core 0 halted; pc=0x02000810
arm7tdmi-s[0,h] % rr
r0=0x00000000 r9=0x00000000 r1_usr=0x05000000 r10_usr=0x00000000
r1=0x05000000 r10=0x00000000 r2_usr=0x0100000c r11_usr=0x00000000
r2=0x00100000c r11=0x00000000 r3_usr=0x02002754 r12_usr=0x020029a0
r3=0x02002754 r12=0x020029a0 r4_usr=0x05000000 sp_svc=0x01002e54
r4=0x05000000 sp=0x01002e54 r5_usr=0x00000000 lr_svc=0x02000818
r5=0x00000000 lr=0x02000818 r6_usr=0x00000000 spsr_svc=0x00000010
r6=0x00000000 pc=0x02000810 r7_usr=0x00000000
r7=0x00000000 cpsr=0x60000053 r8_usr=0x00000000
r8=0x00000000 r0_usr=0x00000000 r9_usr=0x00000000
arm7tdmi-s[0,h] %
Core 1 halted; pc=0x04000000
arm7tdmi-s[0,h] %
Core 1 software breakpoint hit; pc=0x04000c58
arm7tdmi-s[0,h] %
Core 1 running
arm7tdmi-s[0,h] %
Core 1 halted; pc=0x04000c4
arm7tdmi-s[0,h] % mr 0x0500010C
0x0500010c: 0x00000000
arm7tdmi-s[0,h] % mw 0x0500010C 0xC0000000
arm7tdmi-s[0,h] %
Core 1 software breakpoint hit; pc=0x04000c58
arm7tdmi-s[0,h] %
Core 1 running
arm7tdmi-s[0,h] %

```

NOTE: In a real application, shared peripherals and GPIOs are assigned to ACPU in a slightly different manner. Both CPUs have access to a mailbox peripheral with which they send messages to each other. The WCPU has FW (WFW) that listens for and processes these messages. One such message that the ACPU sends is to request a shared peripheral. Another such message is to request a shared GPIO.

JTAG DEBUGGING FROM FLASH

To test the JTAG connection, we used the *Blinky* sample project running from RAM. Now we will attempt to burn an image the compiled code into the flash and debug from there.

1. Start with the Probe and the DB both powered off.
2. Power on the DB.
3. Power on the Probe.
4. On the PC, open a terminal program. Open a telnet connection to the Probe IP address.
5. Reset WCPU by entering the command: `tr`.
6. Read SHCNTL register: `mr 0x05000108`

On reset, all of the shared peripherals are assigned to the WCPU. The SHCNTL register, Shared Peripheral Control, shows this since its value will be `0x00000000` (all bits are 0 = peripheral assigned to WCPU). *MULTI* is configured to talk to Core 1 (ACPU), so in order to program the APP flashes (AF0 and AF1), the ACPU needs control of them.

7. Dump the first few words of AF0 to ensure that the flash is blank: `md 0x06000000`.
If the value read is anything other than all `0xFFFFFFFF`, then the flash is not blank, and it must be erased before being programmed.
8. Erase AF0 by setting the FCDMERASE bit of AF0's register FCERASECNTL. Also set FCERASEVERIFEN to enable flash erase verification: `mw 0x07000300 0x22000000`.
Only the CPU to which the AF0 is assigned can erase it, which, in this case, is the WCPU (Core 0). It is also possible to erase it using ACPU (Core 1) after reassignment.
9. Re-read FCDMERASE to verify flash erase success: `mr 0x07000300`.
This read should return `0x02000000`, signifying a successful erasure.
10. Assign APP Flash 0 and 1 to ACPU: `mw 0x05000108 0x00000180`.
11. Assign GPIO30 and GPIO31 to ACPU: `mw 0x0500010C 0xC0000000`.

```

192.168.3.106 - Tera Term VT
File Edit Setup Control Window Help

Green Hills Probe and SuperTrace Probe for ARM/XScale, v1.12.2
Firmware built Sep 5 2006 12:16:09
Serial No: 8958
Part number 520-GP300-02A
Green Hills Software, Inc
http://www.ghs.com

Type 'info' to list probe information.
Type 'setup' to set the current configuration.
Type 'help' to list the online help.

GreenHills Probe
arm7tdmi-s[0,f] % tr
arm7tdmi-s[0,h] %
Core 0 halted; pc=0x02000810
arm7tdmi-s[0,h] %
Core 1 halted; pc=0x00a67d4c
arm7tdmi-s[0,h] % mr 0x05000108
0x05000108: 0x00000000
arm7tdmi-s[0,h] % md 0x06000000
0x06000000:
0600_0000 ea000007 eaffffff eaffffff eaffffff .....
0600_0010 eaffffff eaffffff e59ff000 eaffffff .k...>.$...$.
0600_0020 00006b8c e329f0d3 e59fd024 e59fc024 ..\.....0.
0600_0030 e35c0000 1a000000 e59fc01c e59f301c
arm7tdmi-s[0,h] % mw 0x07000300 0x22000000
arm7tdmi-s[0,h] % mr 0x07000300
0x07000300: 0x02000000
arm7tdmi-s[0,h] % mw 0x05000108 0x00000180
arm7tdmi-s[0,h] % mw 0x0500010C 0xC0000000
arm7tdmi-s[0,h] % █
    
```

12. Start *MULTI*. This opens MULTI Launcher.
13. Open Blink\blink_G31.gpj GHS project: File → Open Project Builder...
14. Right click on src\blink.gpj GHS project and select Add File Into blink.gpj...
15. Add standalone_romrun.ld.
16. Remove standalone_ram.ld by selecting it then selecting Edit → Remove standalone_ram.ld.
17. Rebuild Blink\blink_G31.gpj (F7).

```

Build Details
Building blink
Compiling main.c because blink.gpj has changed
"src\main.c", line 35: warning #111-D: statement is unreachable
    return 0;
    ^
Linking blink because main.o has changed
Done
Build successful
0 Errors - 1 Warning
    
```

The Blink application image has now been linked to run from ROM (which, in the GS1010 case, is from the flash) as opposed to from RAM.

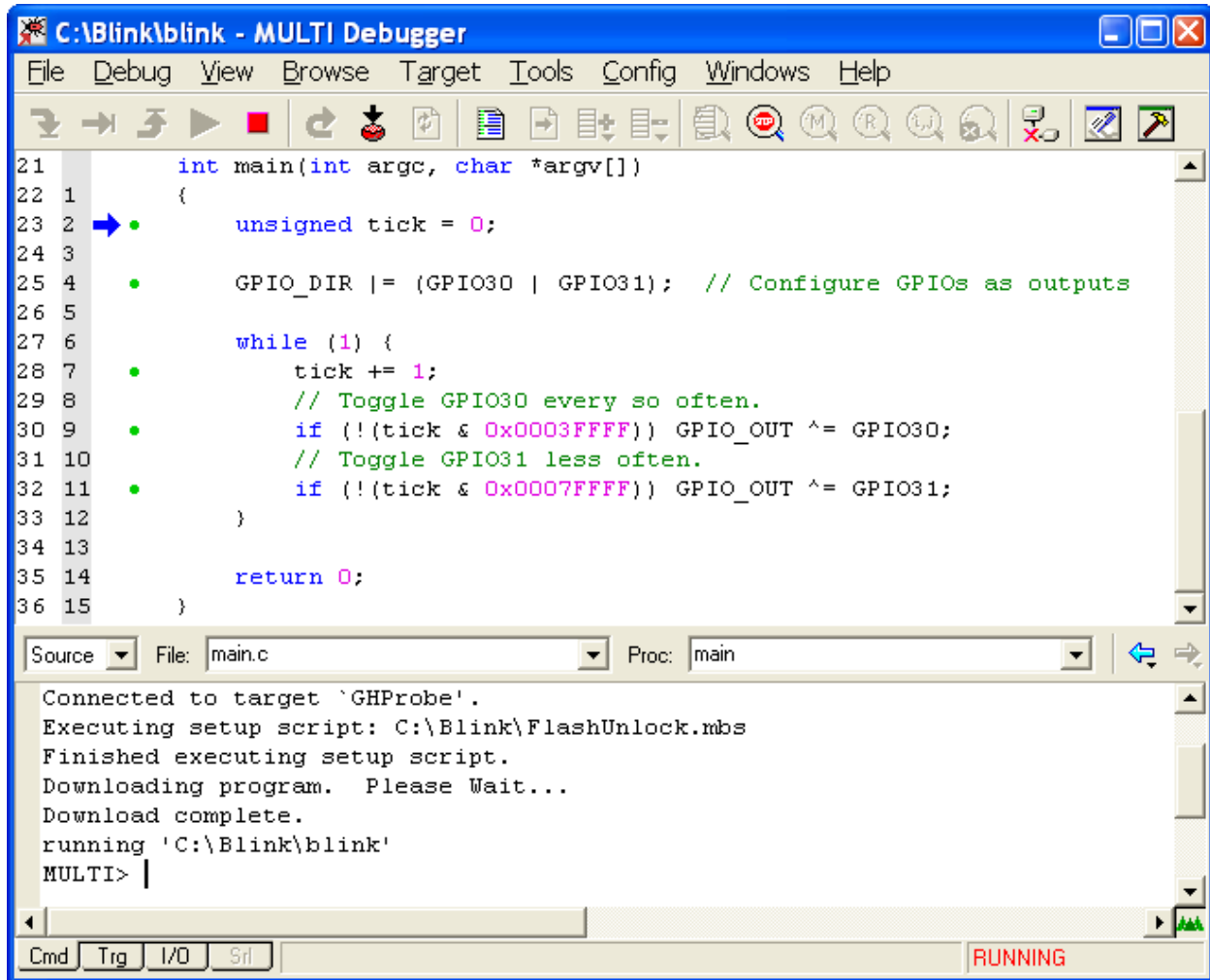
18. From the MULTI Project Builder, connect to the Probe: (F4)
19. Select GHProbe and Download (Download and debug application), and then click Connect.

20. From MULTI Project Builder, select the Project file Blink\blink_G31.gpj and debug it: F5.

21. Run the program: F5.

Running the program will cause *MULTI* to connect to Core 1 and download the compiled code directly into the flash. It then starts the CPU running.

Congratulations... D6 and D7 should now be blinking!!!



```
C:\Blink\blink - MULTI Debugger
File  Debug  View  Browse  Target  Tools  Config  Windows  Help
[Icons]
21      int main(int argc, char *argv[])
22 1    {
23 2    → •   unsigned tick = 0;
24 3
25 4    •   GPIO_DIR |= (GPIO30 | GPIO31); // Configure GPIOs as outputs
26 5
27 6    while (1) {
28 7    •   tick += 1;
29 8    •   // Toggle GPIO30 every so often.
30 9    •   if (!(tick & 0x0003FFFF)) GPIO_OUT ^= GPIO30;
31 10   •   // Toggle GPIO31 less often.
32 11   •   if (!(tick & 0x0007FFFF)) GPIO_OUT ^= GPIO31;
33 12   }
34 13
35 14   return 0;
36 15   }
```

Source File: main.c Proc: main

```
Connected to target 'GHProbe'.
Executing setup script: C:\Blink\FlashUnlock.mbs
Finished executing setup script.
Downloading program. Please Wait...
Download complete.
running 'C:\Blink\blink'
MULTI> |
```

Cmd Trg I/O Sil RUNNING

CONCLUSION

The Probe can be used with the DB to debug the both the WCPU and the ACP. Due to the many different sleep and run modes, however, great care must be taken to understand exactly when the processors are available for debug.



GainSpan Corporation • 121 Albright Way • Los Gatos, CA 94032-1801 • U.S.A.
+1 (408) 689-2129 • info@GainSpan.com • www.GainSpan.com

Copyright © 2008 by GainSpan Corporation. *All rights reserved.*

GainSpan and GainSpan logo are trademarks or registered trademarks of GainSpan Corporation.
Other trademarks are the property of their owners.

Specifications, features, and availability are subject to change without notice.

080125TE