Embedded
Systems
Course

EEE3074W

Lab Practical 03:
Introduction to CSB337 and ARM assembly

Team	Two
Duration	2-3 hours

1 *Introduction*

In this practical you will:

1. Set up the Cogent CSB337 board
2. Compile a simple assembly program for the CSB337
3. Upload the executable to the CSB337
4. Run the program

Condensed into four points, this practical may sound rather simple – but, you can be rest assured that each of these four steps hide plenty of tricky details.

Outcomes of this practical are:

- Set up the CSB337 under Windows:
  - Know what cables to use and how to connect the CSB337 to a PC
  - Set the board rate
  - Setup the local subnet
  - Set the IP number of the CSB337
  - Store this information in the ESAOA framework
- Compile a simple program for the CSB337
  - Run "mm" (the make makefile utility) to generate a makefile for the embedded application
  - Use the gcc-arm-elf cross-tools within the ESAOA framework to build the application
- Uploading and running the program on the CSB337
  - Using xmodem
  - Using tftp
- Saving output results

1.1 Rules

- Arrive at the lab at any time during the assigned practical times, so long as you submit your results, do the demo and the oral exam before the end of the session. However, you are encouraged to arrive when the session starts so that you will have enough time to complete the prac.

- Sign in when you arrive
- Work as a team of two if possible: exceptions to this rule can be approved by the tutor.
- Use your class notes and textbooks during the session if you want to -- but they probably won't help much.
- Ask questions if you get stuck
- Don't leave before you complete the demo and oral exam and have been signed out (if you aren't signed out you may be deemed responsible for missing equipment)

1.2 Demonstration

You need to show you got the application running on the board.

2 *Introduction to the CSB337*

In this practical we will be using the CSB337 from Cogent. It is a single board computer (SBC), as it has a microcontroller and a set of ASICs which provide additional peripherals, such as the Intel flash memory chip that is clearly seen in the photo on the right (see Illustration 1). The AT91RM9200 ARM9 microprocessor is used on the CSB337, in the photo it is the ATMEL chip to the right of the Intel chip. A battery (top right) is also provided which powers the real-time clock (RTC).

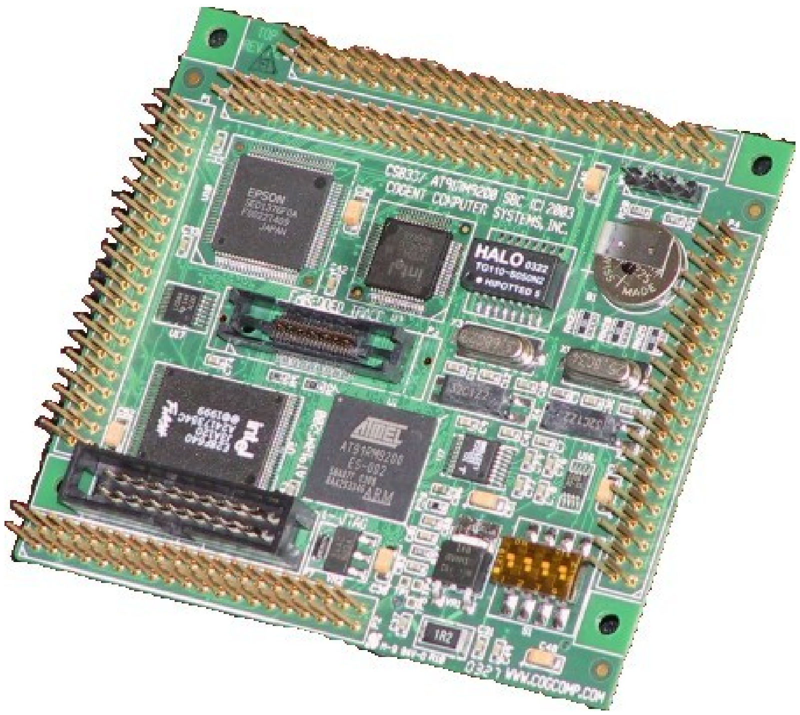


Illustration 1: CSB337

The CSB300CF breakout board has been attached to the CSB337 main board (shown in Illustration 2). The CSB300CF provided provides convenient ports, a few LEDs and push buttons which can be used to test your embedded software.

Do not yet connect up the power. The RS232 serial cable needs to be connected to the breakout board as illustrated in Illustration 2; note that you need to use the bottom port (labeled SERIAL0) that is closest to the +5V power input connector. Connect the Ethernet cable from the lab PC's second Ethernet port (the one that is not connected to the network) to the CSB337's network port that is labeled 10/100 ENET. The network cable must be a "crossover" cable which swaps the TX and RX lines; this is why there is an 'X' on the cable in the photograph. I find it better to connect up the power once you've got the terminal running; this is done in the activities section below.

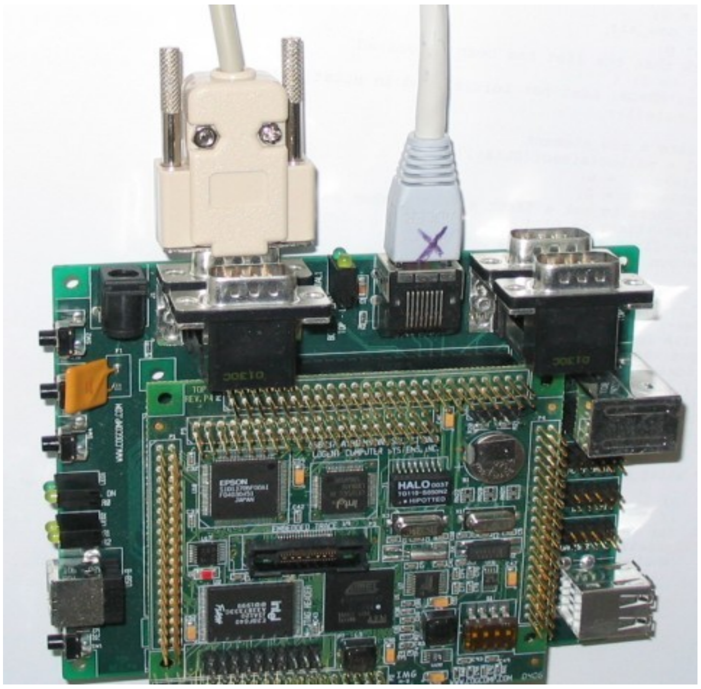


Illustration 2: Breakout Board

3 Activities

3.1 Sign In

Sign in and make sure you have all the needed equipment

Item	Quantity
Cogent CSB337 Evaluation Board	1
Power supply for CSB337	1
RS232C serial cable	1
Ethernet crossover cable	1

3.2 Connecting up the CSB337

Don't connect the power until you have HyperTerminal or other serial terminal program running. This will allow you to see the startup sequence.

3.2.1 Communicating via RS232

RS232 settings

Setting	Value

Baud rate	38400 bps
Parity	None
Data bits	8
Stop bits	1
Flow control	None

Turn on the CSB337 by plugging in the power and see if you get the start-up message. You should see:

```
TFS Scanning //FLASH/...
EMAC: Auto-Negotiate Complete, Link = 100MBIT, Full Duplex.
MICRO MONITOR
CPU: AT91RM9200 ARM920T
Platform: Cogent CSB337 - AT91RM9200 SBC
Built: Feb_02,2004 @ 15:43:20
Monitor RAM: 0x20000000-0x2001a044
Application RAM Base: 0x20100000
MAC: 00:23:31:37:01:18
IP: 192.168.0.2
```

Q1 – What is your CSB337's MAC address?

3.2.2 MicroMonitor commands

The entire MicroMonitor user manual is in $H/Software/Documentation/MicroMonitor/Manual. But you shouldn't need to use it in this practical.

You can list the MicroMonitor commands using the help command (i.e. type "help" at the UMON> prompt in HyperTerminal).

NOTE

Don't mess with the *flash* command or do something like *tfs clean* because this can easily ruin the MicroMonitor installation and it's not easy to fix.

Here's a list of useful MicroMonitor commands:

Command	Description
arp <ip>	Resolve an IP address, can be used like ping to test the network connection.
flash	Don't fiddle with this command or you may wreck the MicroMonitor installation.
heap	Display heap information
help [cmd]	List commands or show more detailed help on a specific command 'cmd'.
mstat	Very useful: displays current status.
reset	Trigger a software reset

set	Set environment variables – don't use yet or you might mess things up!
sleep <seconds>	Delay for *seconds* seconds
tfs	Tiny File System (TFS) commands, used to manage files in flash.
tftp	Trivial FTP for uploads/downloads
xmodem	Transfer data over serial connection

Q2 – What MicroMonitor command (and it is not '*flash*') can you use to list the file directory in flash?

3.2.3 Setting up MicroMonitor's Ethernet

Set up the network for the CSB337 using the *set* command. Type set to view the current settings. If the IPADD for your CSB337 isn't already set up 192.168.0.2, then change it using the set command as follows:

```
set IPADD 192.168.0.2
```

Do the same for the others if the are different or haven't been not set. Make sure that you have the same settings for the variables shown below:

```
APPRAMBASE = 0x20100000
IPADD = 192.168.0.2
NETMASK = 255.255.255.0
DHCPLEASETIME = 0x8051010
HOSTIPADD = 192.168.0.1
RAMDISKBASE = 0x20800000
```

Once you are happy with the settings, save them using:

```
set -f monrc
```

And then enter a reset command (or press the reset button, SW1) to reboot the CSB337. This has to be done in order for the system to load the new settings.

3.2.4 Testing the network communication

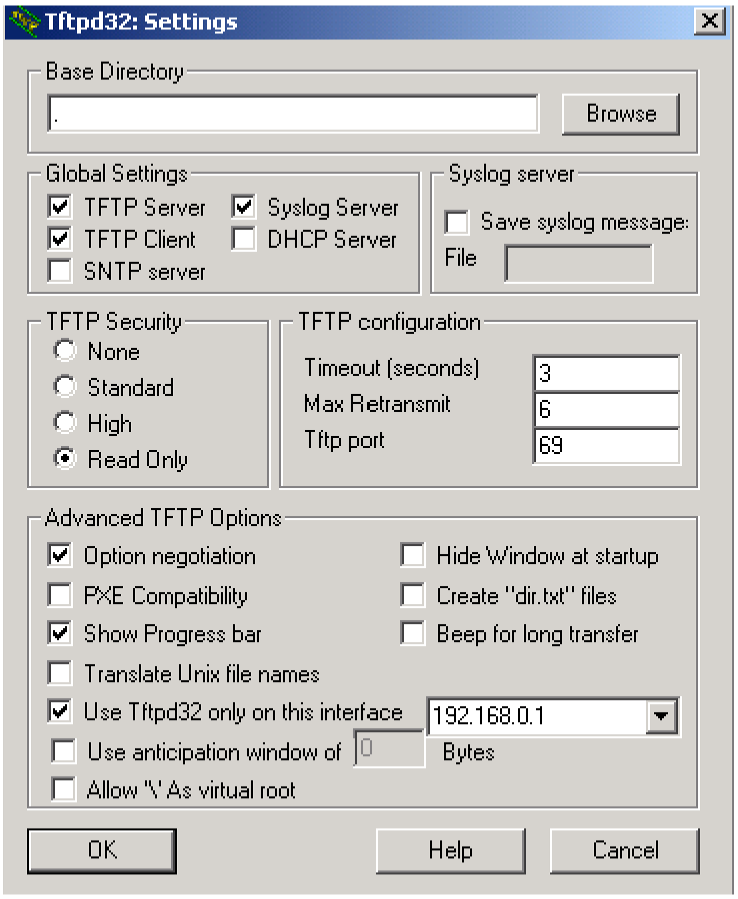
Q3 – What MicroMonitor command did you use to test the subnet?

3.2.5 Installing a TFTP server

Windows usually doesn't include a TFTP server. But we need to use one. A safe tftp server is located in directory \\forge.ee.uct.ac.za\EEE374W\Software\. You should be able to open this link in Windows explorer. Copy the whole TFTPD directory to your desktop (this will make it run faster).

Next, double-click the tftpd32 icon. It will complain about privileges: just press OK. Click on the settings button to open the settings. Change the settings as shown on the right. Make sure the DHCP server is disable and that the correct IP address is being used.

Close the tftpd application and rerun it. This will make the new settings take effect, and you shouldn't get the error message.



Base Directory: .

Global Settings:
- ✓ TFTP Server
- ✓ Syslog Server
- ✓ TFTP Client
- ☐ DHCP Server
- ☐ SNTP server

Syslog server:
- ☐ Save syslog message
- File:

TFTP Security: None / Standard / High / [✓ Read Only]

TFTP configuration:
- Timeout (seconds): 3
- Max Retransmit: 6
- Tftp port: 69

Advanced TFTP Options:
- ✓ Option negotiation
- ☐ Hide Window at startup
- ☐ PXE Compatibility
- ☐ Create "dir.txt" files
- ✓ Show Progress bar
- ☐ Beep for long transfer
- ☐ Translate Unix file names
- ✓ Use Tftpd32 only on this interface: 192.168.0.1
- ☐ Use anticipation window of 0 Bytes
- ☐ Allow '\' As virtual root

3.2.6 Testing upload via TFTP

Login to forge and run the following commands:

```
$ enter-aoa
$ cd Projects
$ tar -zxf /EEE374W/Pracs/Prac02/Prac02.tar.gz
```

[Yes, this is the correct filename]

Open directory *\\forge.ee\home\username\aoa\Projects\Prac02\Software\Applications\Test* and copy the file Test.cm.elf into your TFTPD directory on your desktop. We are going to put this file on the CSB337 to test that the board is working. You could use *scp* or WinSCP or connect to the server using Nautilus [Gnome desktop file browser] or similar. This will also allow you to directly edit the source files on the server (allowing easier compilation).

Now, from MicroMonitor enter the command:

```
tftp -fE -FTest.cm.elf 192.168.0.1 get Test.cm.elf
```

If you look at the TFTP Server log window in TFTPD32, you will see that a connection was received from 192.168.0.2 on some port, and that the file Test.cm.elf was sent.

On MicroMonitor, you should see a message "Retrieving Test.cm.elf ..." show up. A red LED should light up on the board to show that it is busy writing to the flash.

NB: Do NOT disconnect the CSB337 or press reset while the red LED is on, because this indicates that the device is busy writing to flash.

Disconnecting while this is happening results in possible corruption of your flash memory.

3.3 Login and obtain Prac03.tar.gz

Login to forge.ee.uct.ac.za using your username.

Run the following commands:

```
$ enter-aoa
Change directory into projects
$ tar -zxf /EEE3074W/Pracs/Prac03/Prac03.tar.gz
$ cd Prac03/Software/Applications/Basic
$ mm
$ m clean
$ m
```

This will print out a long list of output, including showing that the "asm" files have been compiled.

3.4 ARM Assembly Language

You will find a guide for using ARM assembly in directory /EEE374W/Documentation/Software/GAS .

A few points about the document, which you may want to have open while working on the assembler task:

- Invoking the Assembler: skip, because this is done in the make file. Want Details? If so, then see files config.make and defs.make in directory PDM/CSB337/MicroMonitor.
- Assembly Language Syntax: pretty obvious, just skim.
- Assembler Directives: Have a look at .asciz , .byte, .data, .equ, .extern, .global, .text, .word, and .include, but *NOT* **.end** because it just causes problems if you want to include modules in other modules.
- The rest you can pretty much skip over .

The document "/EEE374W/Documentation/Hardware/QuickRef/ARM-Instructions-QuickRef.pdf" is much more useful, assuming you know the GAS syntax. A suggestion is to jot down instructions (like MULS that you think may be useful in the tasks below.

3.5 Assembly Routine: your_strlen

For this task, you need to develop a string length routine, as in standard C, that calculates the number of characters in a null-terminated string, excluding the Null terminator. You'll probably want to use the ARM quick reference manual to look up commands to use.

Files to edit: Go into directory $PDM/CSB337/MicroMonitor, and open up **asm_meths.h** and **asm_meths.S**. Implement the assembly function ***your_strlen*** in asm_meths.S, search for TODO[1] in the file. Try to make it more optimal (hint: you may need to push a few more registers on the stack).

<u>NOTE</u>: Search for AUTHORS in the file "asm_meths.S" and add your and your team mate's names and student numbers into this comment section.

But wait, if you think this is a totally daunting task, from the terminal, "go app b" and then do a "m asm", which will generate assembly output for a variety of modules. One of particular interest, the module **Utils/Stream.S**, happens to contain a function called **my_strlen**, which may give you a convenient, although not optimal, implementation for strlen. *Undefine* the macro called **USE_YOURS** in **init.c** in order to make the program use the my_strlen function if you want to test *my* code before testing *your* code [1].

3.5.1 Download BIN to CSB337

Compile the application (which assembles the asm_meths.S module) and then use the following method to get the .bin file over to the CSB337 in order to execute the program. It's best to download into RAM, and execute from there, so that you do not have to keep writing the ELF file into flash.

Use the TFTPd program explained earlier.

Execute the following commands on the CSB337 to transfer the file:

```
$ tftp 192.168.0.1 get Basic.cm.bin $APPRAMBASE
$ call $APPRAMBASE
```

PANIC

Q4 – Fill in the statistics for your routine

1 I suppose naming things *my_function* and *your_function* is an example of what *not* to do because it's like totally confusing. A better method would be naming *my_strlen* as *c_strlen* to make better sense.