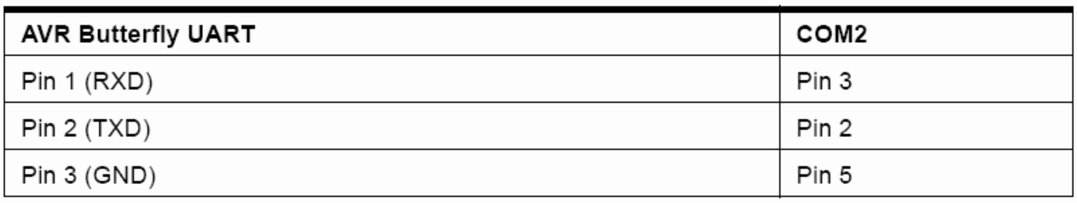
Programming the butterfly for idiots like me

Ajb 10/11/03
This is how I programmed the butterfly for the first time. I MAKE NO GARENTEES, NOR DO I ASSUME ANY LIABILITIES, USE THIS GUIDE AT YOUR OWN RISK, this is just how I did it.
Please send pleasant comments and suggestions to aboehnlein@yahoo.com, and unpleasant ones to /dev/null
mt 4/2004 – small update

Connections

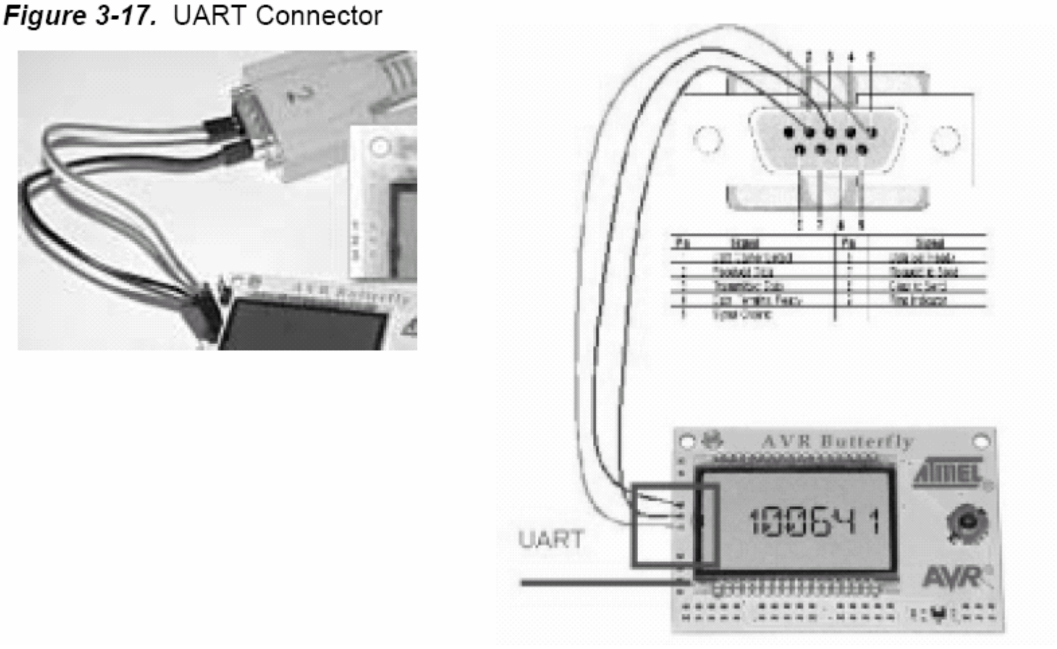
First, you need a communication cable:

Table 3-1. UART



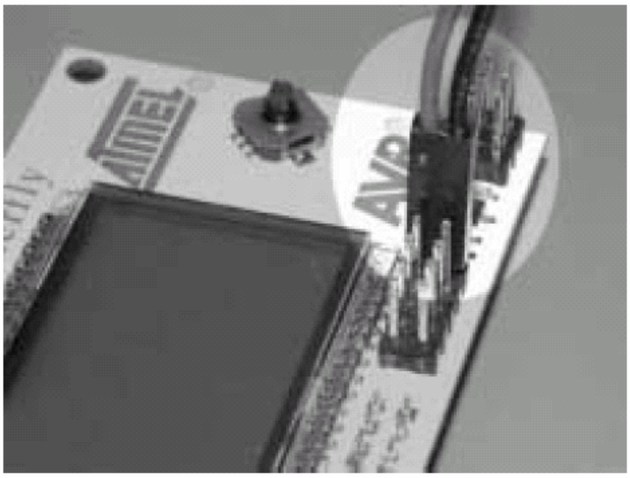
AVR Butterfly UART	COM2
Pin 1 (RXD)	Pin 3
Pin 2 (TXD)	Pin 2
Pin 3 (GND)	Pin 5

Figure 3-17. UART Connector



And a Power Cable:

Figure 3-10. External Power

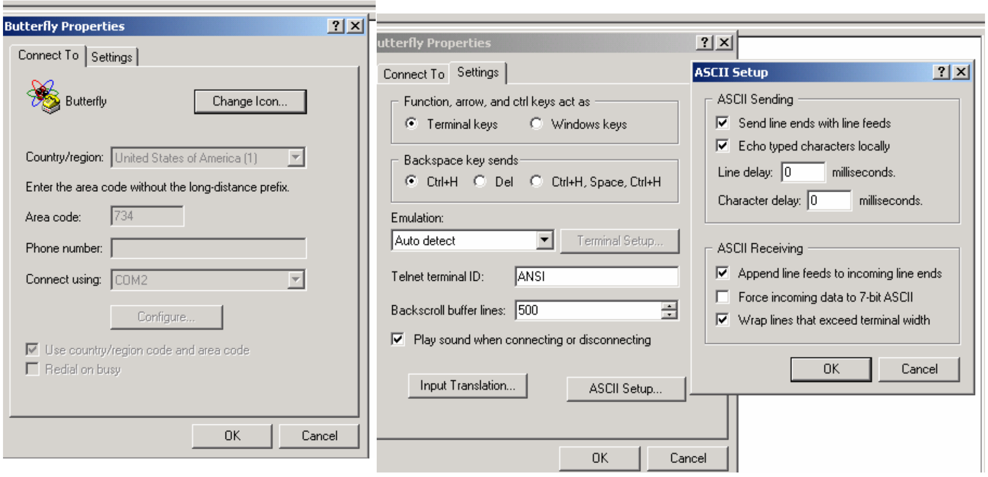


External power can be applied at pin 9 and 10 at both PORTB and PORTD, see *Figure 3-7* for the pinout.

(Remark mthomas: Voltage should be between 3,1 and 4,5V to make sure the battery power supply is turned off and the Butterfly only works from external source. (4/2004) At the ISP-Port there is also a connection for VCC and GND)

Now test your connection to the Butterfly:

Source: AVR Butterfly Evaluation Kit User Guide
http://www.atmel.com/dyn/resources/prod_documents/doc4271.pdf



Butterfly Properties

Connect To | Settings

Butterfly

Country/region: United States of America (1)

Enter the area code without the long-distance prefix.

Area code: 734

Phone number:

Connect using: COM2

☑ Use country/region code and area code

□ Redial on busy

Butterfly Properties

Connect To | Settings

Function, arrow, and ctrl keys act as: ● Terminal keys ○ Windows keys

Backspace key sends: ● Ctrl+H ○ Del ○ Ctrl+H, Space, Ctrl+H

Emulation: Auto detect

Telnet terminal ID: ANSI

Backscroll buffer lines: 500

☑ Play sound when connecting or disconnecting

ASCII Setup

ASCII Sending

☑ Send line ends with line feeds

☑ Echo typed characters locally

Line delay: 0 milliseconds.

Character delay: 0 milliseconds.

ASCII Receiving

☑ Append line feeds to incoming line ends

□ Force incoming data to 7-bit ASCII

☑ Wrap lines that exceed terminal width

Figure: Entering your Name using a terminal-software

Entering Your Name Using a Terminal:

Connect a serial cable from the PC to the AVR Butterfly as described in Section 3.7 “Connect to PC”, Open a terminal on your PC (e.g. HyperTerminal) and configure the terminal to 19200 Baudrate, 8 Databits, none Parity and one stop bit.

Press the joystick up (“SCROLL UP”) to wake the AVR Butterfly. If “AVR BUTTERFLY” is not scrolling over the display, press the joystick to the left (“EXIT SUB-MENU”) until it does. (remark mthomas: “UP” also leaves the bootloader code that is activated after power cycle and reset)

Press the joystick down (“SCROLL DOWN”) three times, so the string “NAME” is displayed.

Press the joystick to the right (“ENTER SUB-MENU”). If this is the first time a name is entered, the string “ENTER NAME” will be displayed, otherwise the name already entered will be displayed and you have to press the joystick to the right (“ENTER SUB-MENU”) once more.

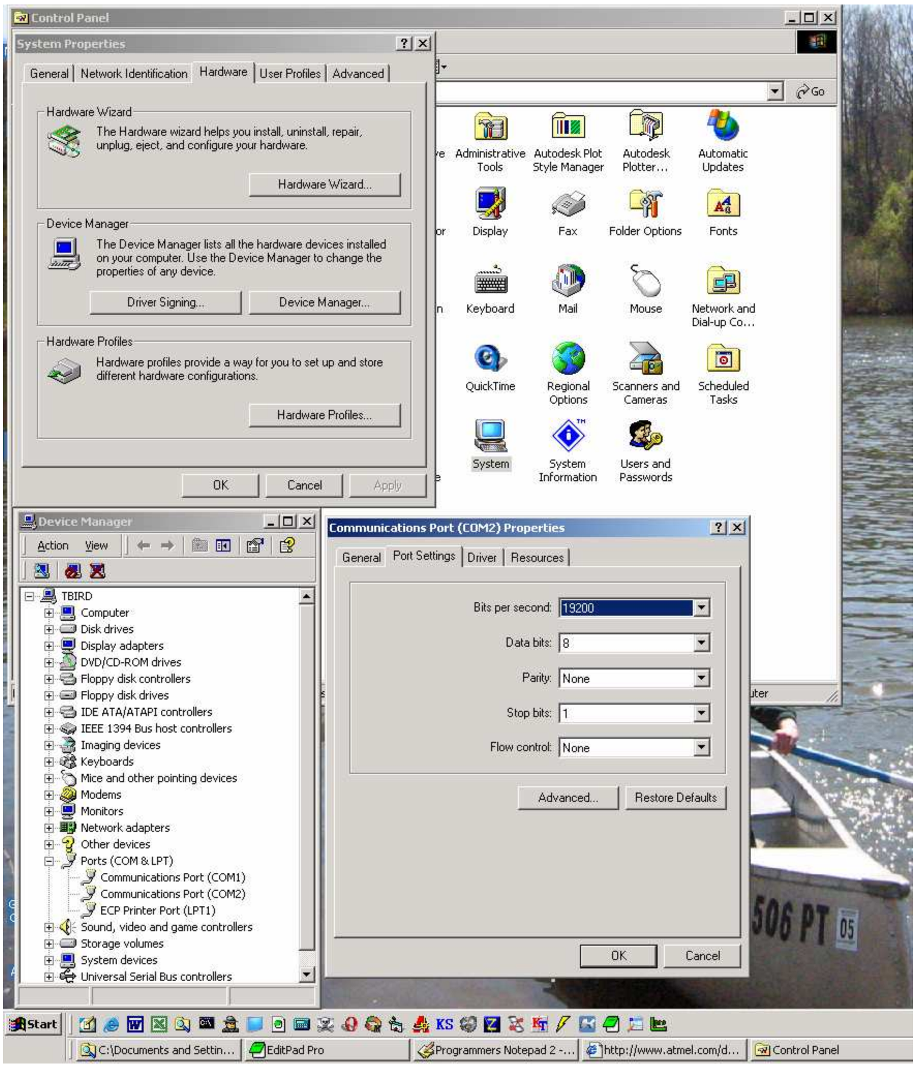
When the “ENTER NAME” is displayed press the joystick down (“SCROLL DOWN”), and “DOWNLOAD NAME” will be displayed 4. Press center push (“ENTER”) to activate the UART, and the text “WAITING FOR INPUT ON RS232” will be displayed.

Type your name in the terminal window on the PC (up to 25 characters) and save the name by pressing enter on your PC-keyboard. The name you typed should now be displayed in the AVR Butterfly display.

Note 1: The Auto Power Off feature is default enabled. It will turn off the LCD after default 30 minutes. This timeout can be changed or turned off. To wake the AVR Butterfly from SLEEP, press the joystick in the UP-position.

Note 2: My computer uses com2 for this, yours may be different.

Set the port using start-/-settings-/-system-/-communications port (com2)-/-port settings



The Hardware wizard helps you install, uninstall, repair, unplug, eject, and configure your hardware.

The Device Manager lists all the hardware devices installed on your computer. Use the Device Manager to change the properties of any device.

Hardware profiles provide a way for you to set up and store different hardware configurations.

Bits per second: 19200

Data bits: 8

Parity: None

Stop bits: 1

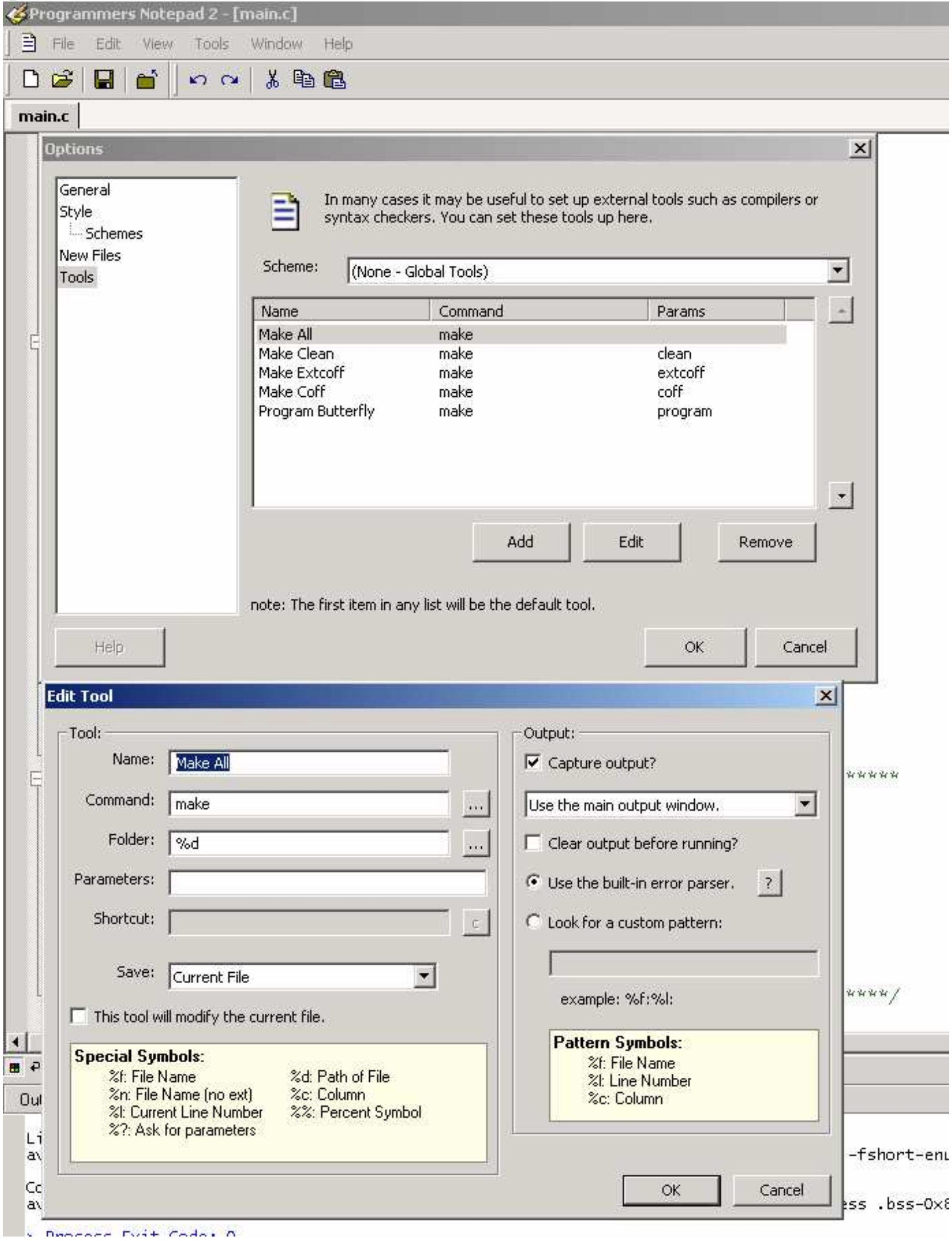
Flow control: None

Figure: Setting up the com-Port (remark mthomas: this step may be unnecessary)

Setup Programmers Notepad

(remark mthomas: Programmers Notepad comes with the WinAVR collection.)

Setup the Tools-Menu of PN



In many cases it may be useful to set up external tools such as compilers or syntax checkers. You can set these tools up here.

Scheme: (None - Global Tools)

Name	Command	Params
Make All	make	
Make Clean	make	clean
Make Extcoff	make	extcoff
Make Coff	make	coff
Program Butterfly	make	program

note: The first item in any list will be the default tool.

Tool:

Name: Make All

Command: make

Folder: %d

Parameters:

Shortcut:

Save: Current File

□ This tool will modify the current file.

Special Symbols:

%f: File Name	%d: Path of File
%n: File Name (no ext)	%c: Column
%l: Current Line Number	%%: Percent Symbol
%?: Ask for parameters	

Output:

☑ Capture output?

Use the main output window.

□ Clear output before running?

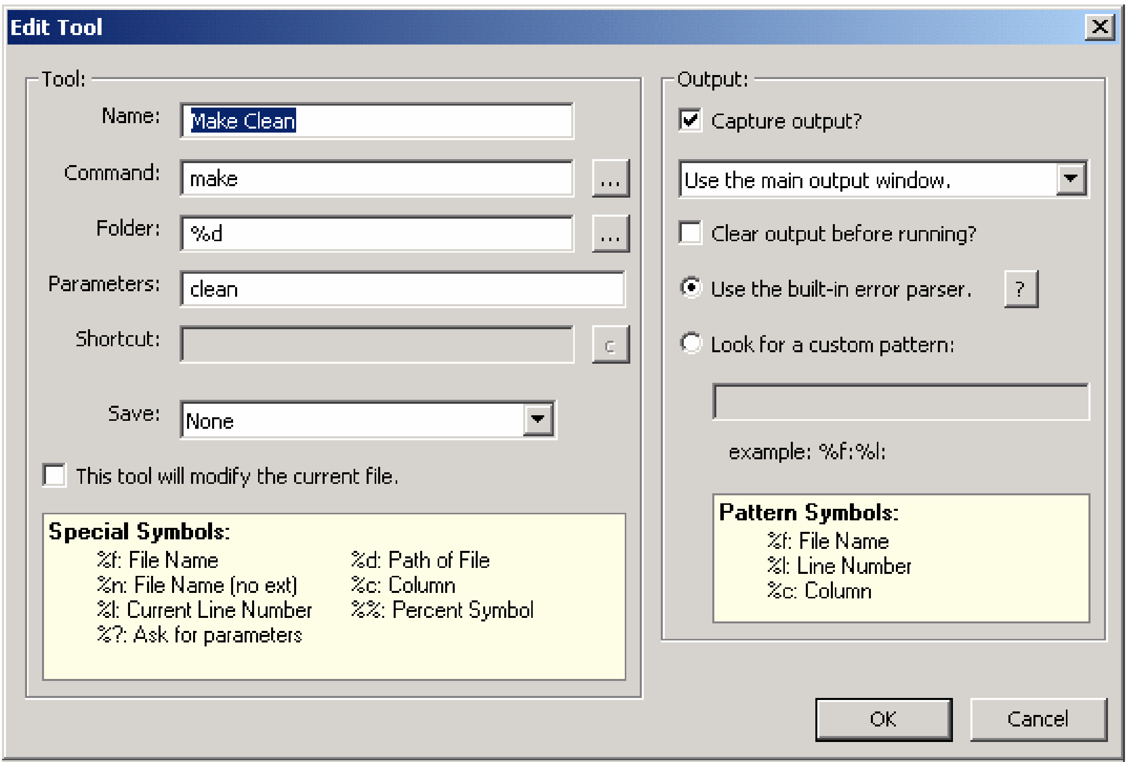
● Use the built-in error parser.

○ Look for a custom pattern:

example: %f:%l:

Pattern Symbols:

%f: File Name
%l: Line Number
%c: Column



Edit Tool

Tool:

Name: Make Clean

Command: make

Folder: %d

Parameters: clean

Shortcut:

Save: None

□ This tool will modify the current file.

Special Symbols:

%f: File Name	%d: Path of File
%n: File Name (no ext)	%c: Column
%l: Current Line Number	%%: Percent Symbol
%?: Ask for parameters	

Output:

☑ Capture output?

Use the main output window.

□ Clear output before running?

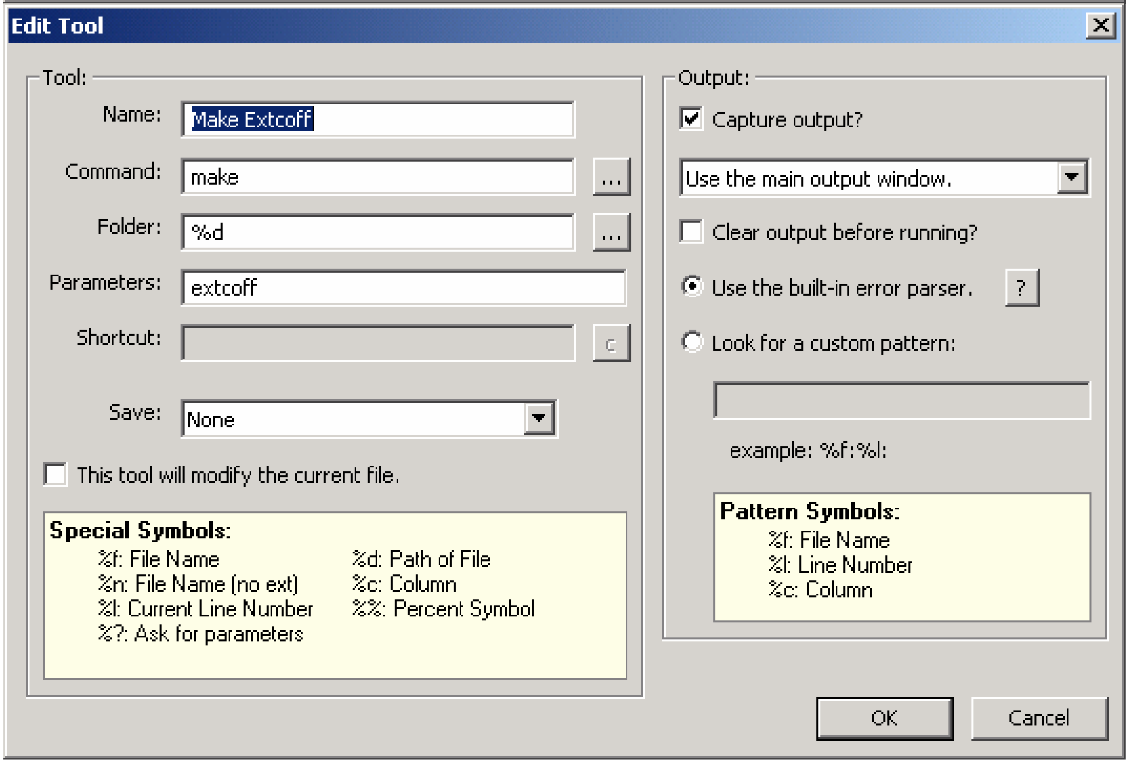
● Use the built-in error parser.

○ Look for a custom pattern:

example: %f:%l:

Pattern Symbols:

%f: File Name
%l: Line Number
%c: Column



Edit Tool

Tool:

Name: Make Extcoff

Command: make

Folder: %d

Parameters: extcoff

Shortcut:

Save: None

□ This tool will modify the current file.

Special Symbols:

%f: File Name	%d: Path of File
%n: File Name (no ext)	%c: Column
%l: Current Line Number	%%: Percent Symbol
%?: Ask for parameters	

Output:

☑ Capture output?

Use the main output window.

□ Clear output before running?

● Use the built-in error parser.

○ Look for a custom pattern:

example: %f:%l:

Pattern Symbols:

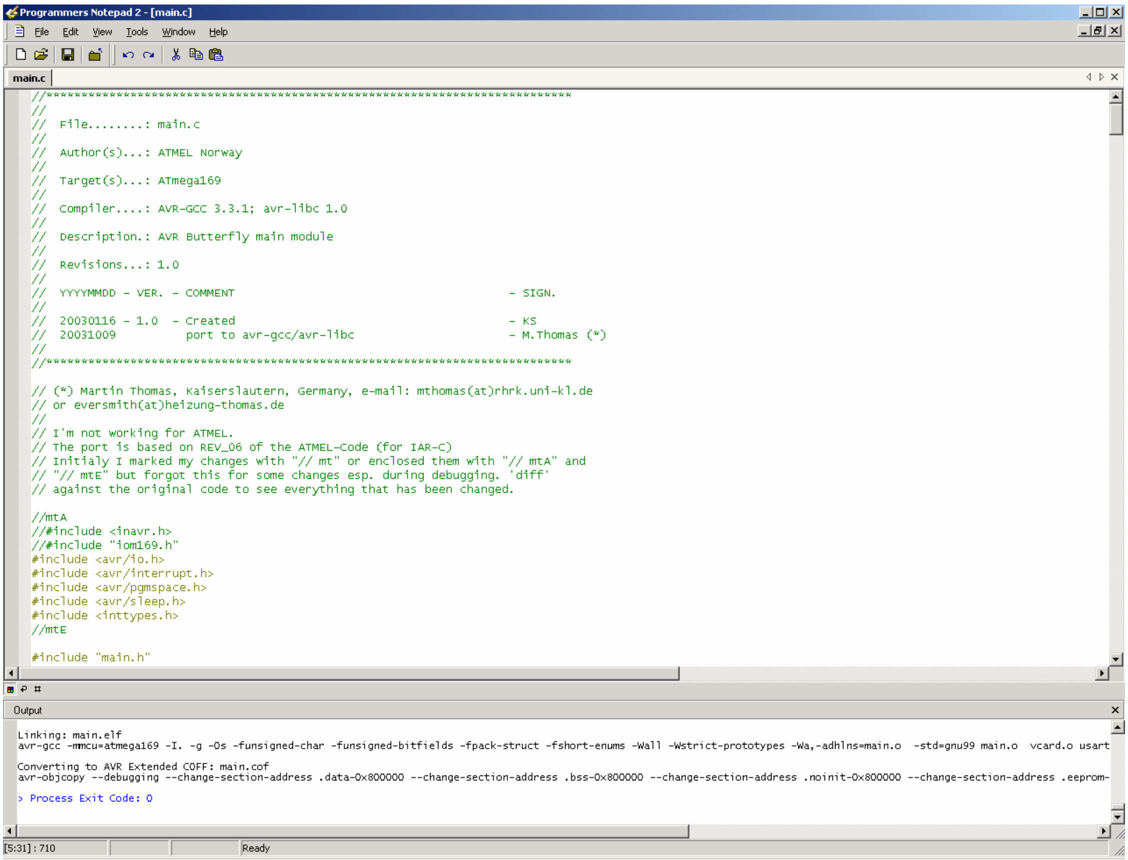
%f: File Name
%l: Line Number
%c: Column

Download the Application Source-Code

Go to http://www.siwawi.arubi.uni-kl.de/avr_projects/ and search the link *Download the Source-Archive (0.6.4/20031205)* (while you are reading this a newer version might be available), right-click, select “save link target” and download the source-archive (bf_gcc_20031205.zip) to your machine. Unpack the Archive.

Compile and Link the code

Open up main.c with programmers notepad and click tools-/- make extcoff
It should look like this….



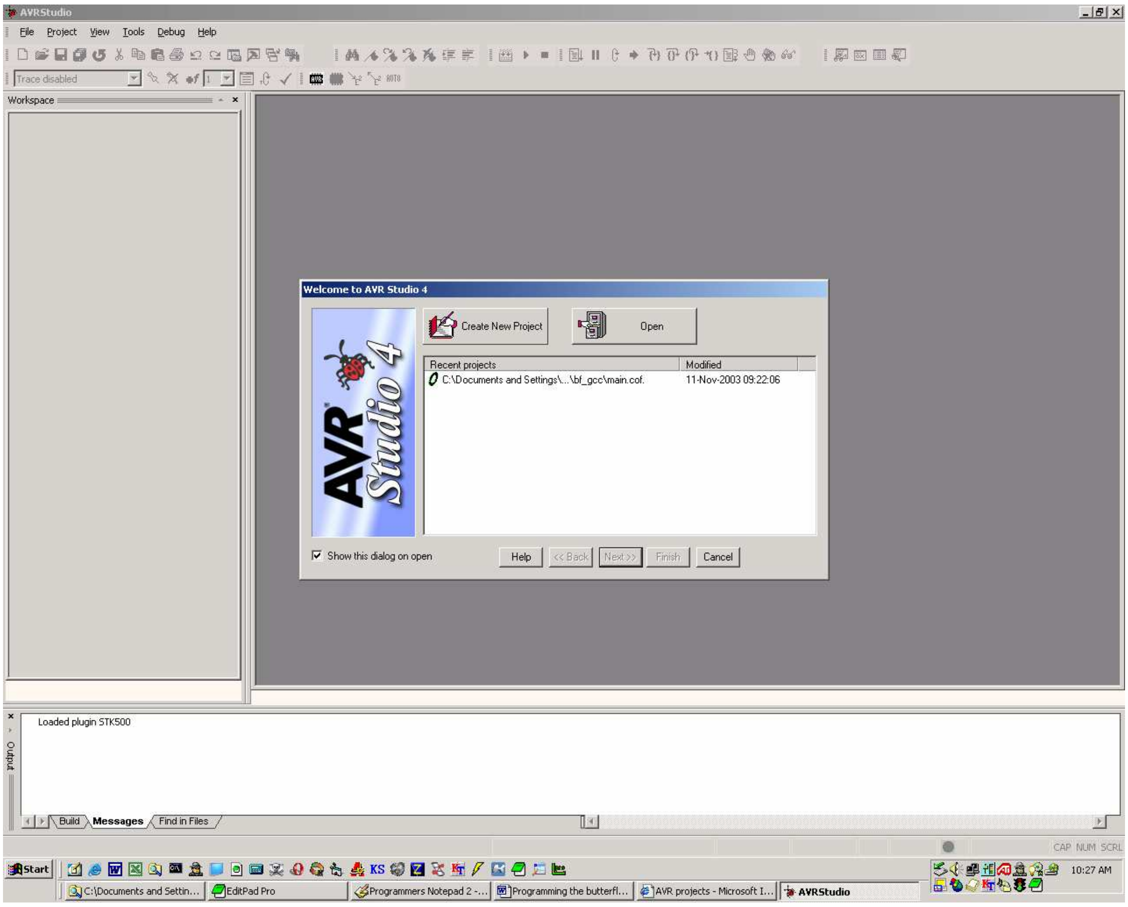
```
//****************************************************************************
//
//  File........: main.c
//
//  Author(s)...: ATMEL Norway
//
//  Target(s)...: ATmega169
//
//  Compiler....: AVR-GCC 3.3.1; avr-libc 1.0
//
//  Description.: AVR Butterfly main module
//
//  Revisions...: 1.0
//
//  YYYYMMDD - VER. - COMMENT                                       - SIGN.
//
//  20030116 - 1.0  - Created                                       - KS
//  20031009          port to avr-gcc/avr-libc                      - M.Thomas (*)
//
//****************************************************************************

// (*) Martin Thomas, Kaiserslautern, Germany, e-mail: mthomas(at)rhrk.uni-kl.de
// or eversmith(at)heizung-thomas.de
//
// I'm not working for ATMEL.
// The port is based on REV_06 of the ATMEL-Code (for IAR-C)
// Initialy I marked my changes with "// mt" or enclosed them with "// mtA" and
// "// mtE" but forgot this for some changes esp. during debugging. 'diff'
// against the original code to see everything that has been changed.

//mtA
//#include <inavr.h>
//#include "iom169.h"
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/pgmspace.h>
#include <avr/sleep.h>
#include <inttypes.h>
//mtE

#include "main.h"
```

```
Linking: main.elf
avr-gcc -mmcu=atmega169 -I. -g -Os -funsigned-char -funsigned-bitfields -fpack-struct -fshort-enums -Wall -Wstrict-prototypes -Wa,-adhlns=main.o  -std=gnu99 main.o  vcard.o usart

Converting to AVR Extended COFF: main.cof
avr-objcopy --debugging --change-section-address .data-0x800000 --change-section-address .bss-0x800000 --change-section-address .noinit-0x800000 --change-section-address .eeprom-

> Process Exit Code: 0
```

Debugging the Butterfly Application with AVR-Studio

(remark mthomas: this step is optional. You may skip this step and continue with section “Programming the Butterfly with the application code“)

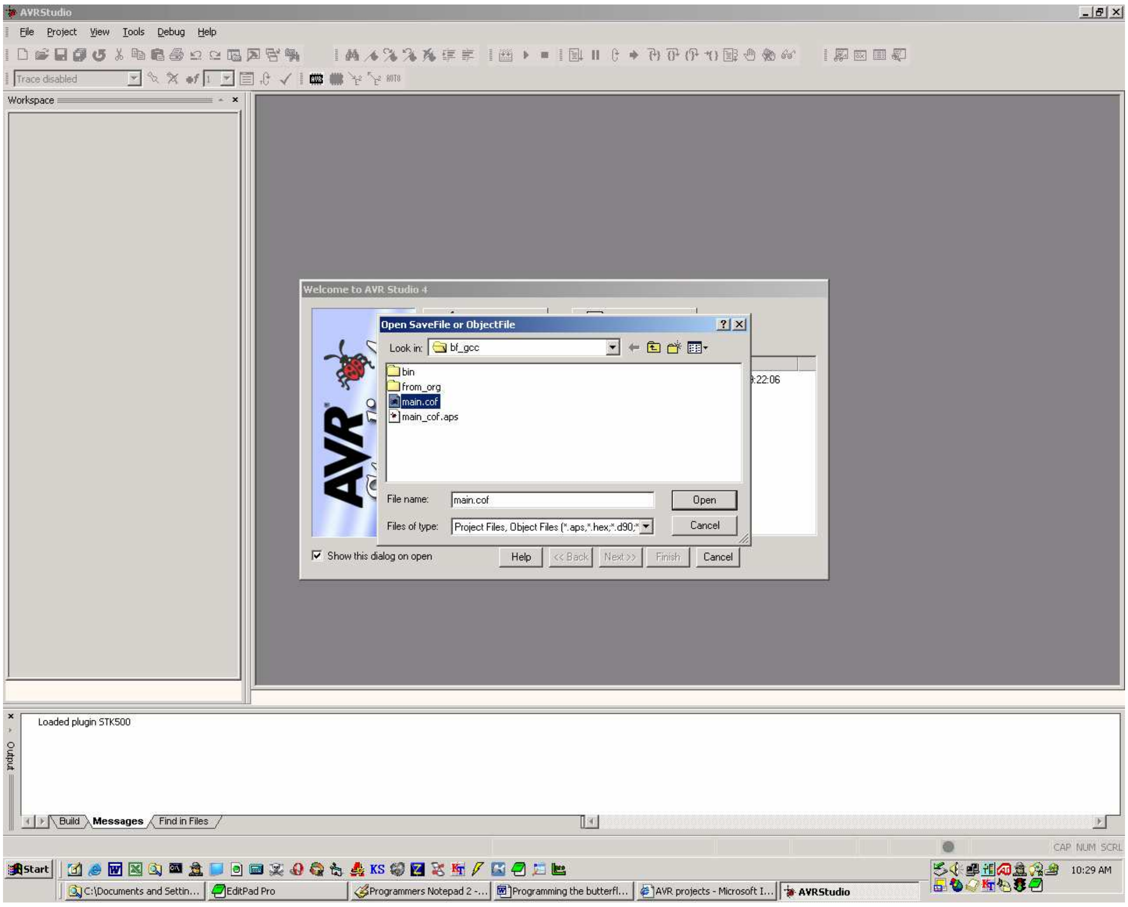
Open AVRStudio and click on open….



Recent projects	Modified
C:\Documents and Settings\...\bf_gcc\main.cof.	11-Nov-2003 09:22:06

☑ Show this dialog on open

```
Loaded plugin STK500
```



Look in: bf_gcc

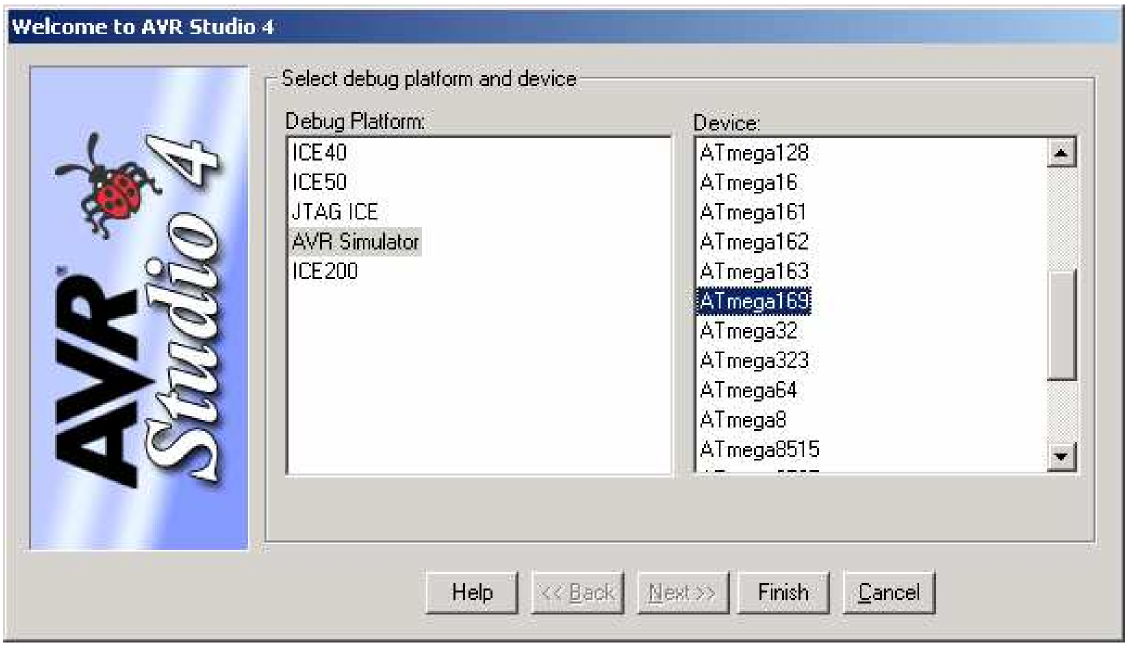
bin
from_org
main.cof
main_cof.aps

File name: main.cof

Files of type: Project Files, Object Files (*.aps,*.hex,*.d90;*

☑ Show this dialog on open

```
Loaded plugin STK500
```



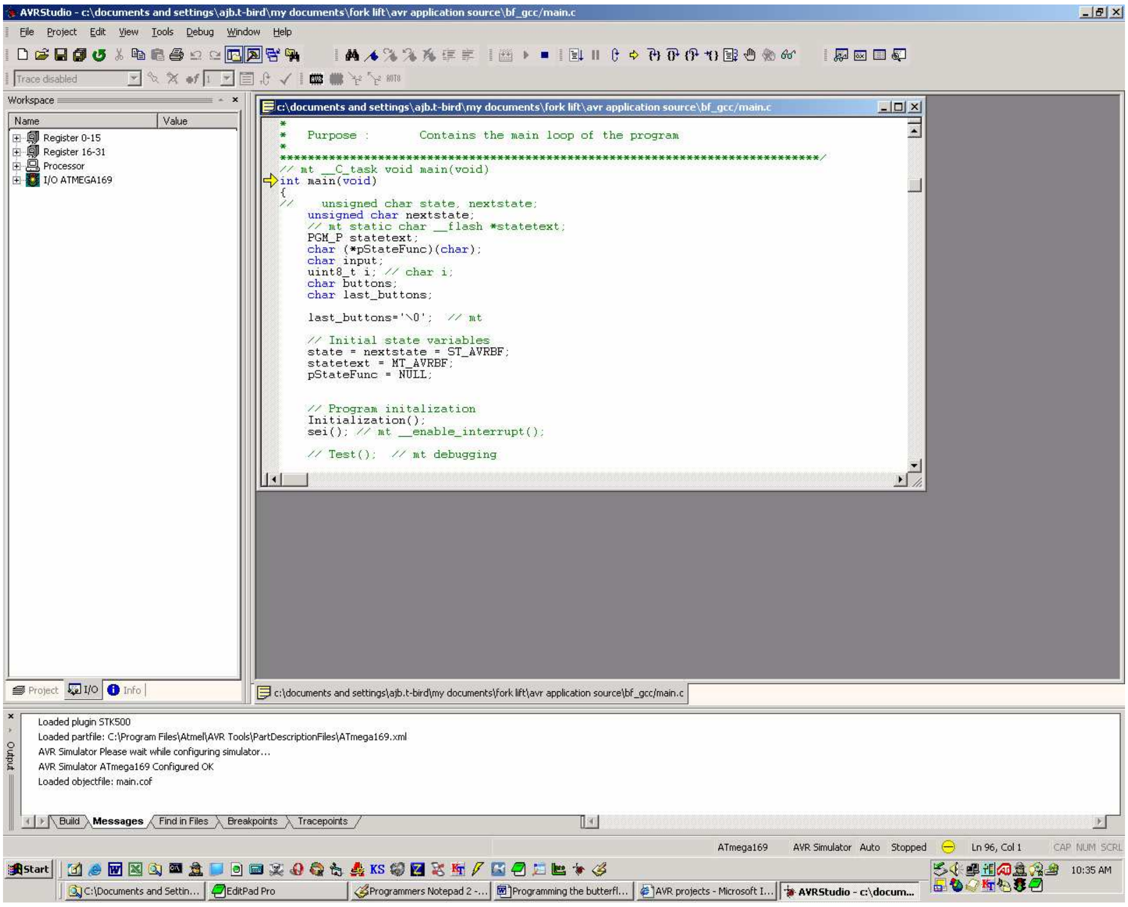
Select debug platform and device

Debug Platform:

- ICE40
- ICE50
- JTAG ICE
- AVR Simulator
- ICE200

Device:

- ATmega128
- ATmega16
- ATmega161
- ATmega162
- ATmega163
- ATmega169
- ATmega32
- ATmega323
- ATmega64
- ATmega8
- ATmega8515


AVRStudio - c:\documents and settings\ajb.t-bird\my documents\fork lift\avr application source\bf_gcc/main.c
File Project Edit View Tools Debug Window Help
Trace disabled
Workspace
Name
Value
Register 0-15
Register 16-31
Processor
I/O ATMEGA169
Project
I/O
Info
c:\documents and settings\ajb.t-bird\my documents\fork lift\avr application source\bf_gcc/main.c
Output
Build
Messages
Find in Files
Breakpoints
Tracepoints
ATmega169
AVR Simulator
Auto
Stopped
Ln 96, Col 1
CAP NUM SCRL
Start
C:\Documents and Settin...
EditPad Pro
Programmers Notepad 2 -...
Programming the butterfl...
AVR projects - Microsoft I...
AVRStudio - c:\docum...
10:35 AM

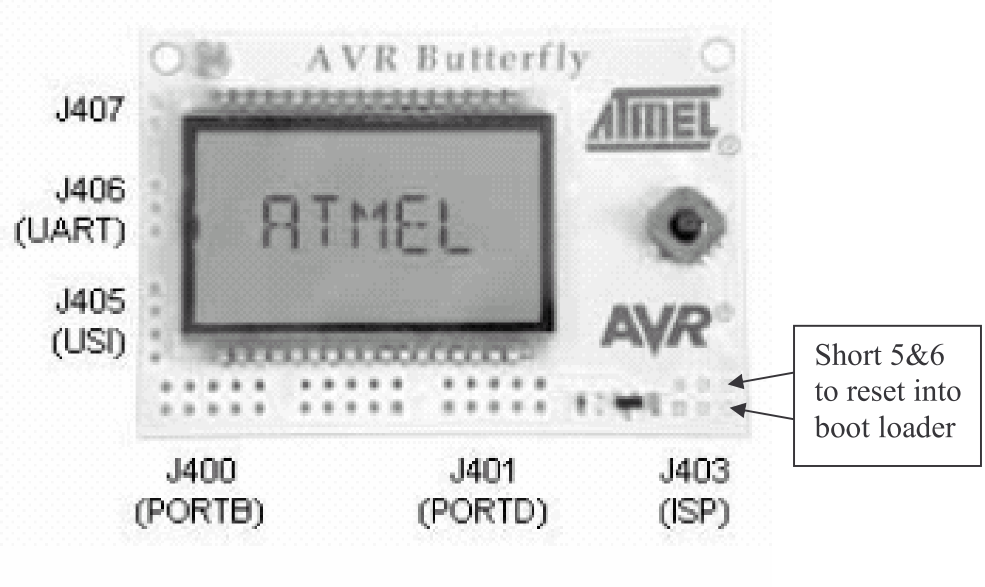
```
*
*   Purpose :       Contains the main loop of the program
*
***************************************************************************/
// mt __C_task void main(void)
int main(void)
{
//    unsigned char state, nextstate;
    unsigned char nextstate;
    // mt static char __flash *statetext;
    PGM_P statetext;
    char (*pStateFunc)(char);
    char input;
    uint8_t i; // char i;
    char buttons;
    char last_buttons;

    last_buttons='\0';  // mt

    // Initial state variables
    state = nextstate = ST_AVRBF;
    statetext = MT_AVRBF;
    pStateFunc = NULL;


    // Program initalization
    Initialization();
    sei(); // mt __enable_interrupt();

    // Test();  // mt debugging
```

```
Loaded plugin STK500
Loaded partfile: C:\Program Files\Atmel\AVR Tools\PartDescriptionFiles\ATmega169.xml
AVR Simulator Please wait while configuring simulator...
AVR Simulator ATmega169 Configured OK
Loaded objectfile: main.cof
```

Programming the Butterfly with the application code

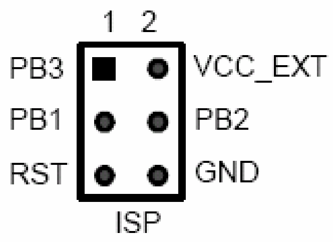
Put butterfly in boot-loader mode

A jump to the boot section can be done with the joy stick, “Options>Boot loader> Jump to Boot loader”, or just reset the ATmega169 by shortcut pin 5 and 6 on J403 the ISP connector, (after a reset the ATmega169 will start in the boot section). (remark mthomas: you may remove the battery and the external power. The Butterfly bootloader is activated after reestablishing the power supply either from battery or external source.)



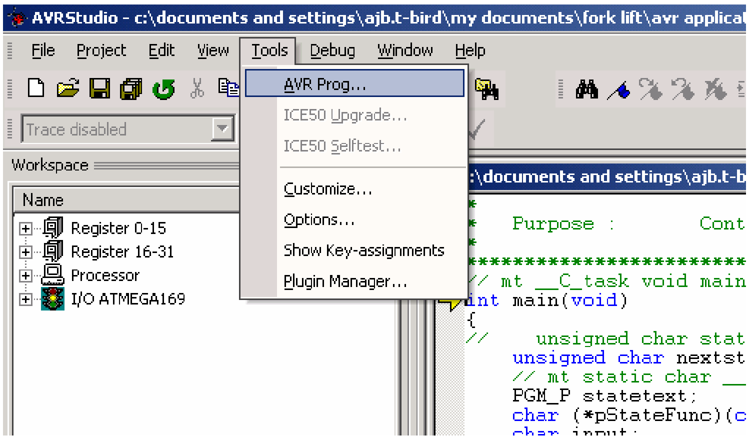
Short 5&6 to reset into boot loader

Figure 3-3. ISP Connector, J403



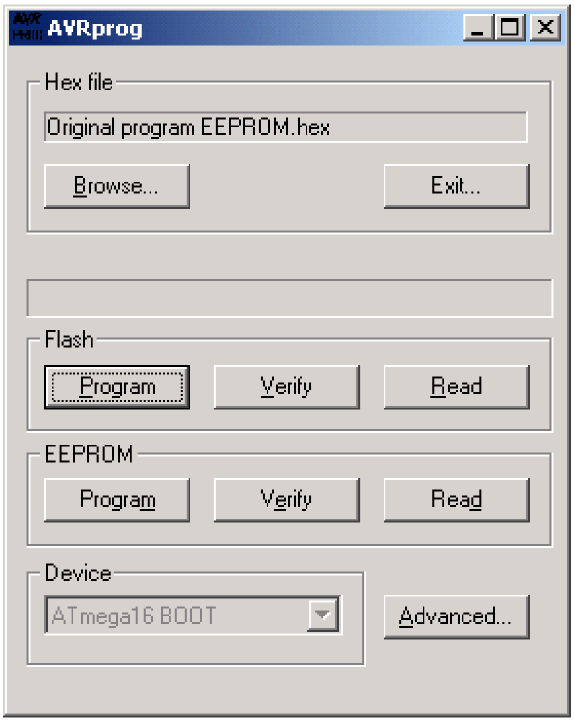
See Figure 3-3 for the pinout of the ISP-connector. Nothing will be displayed on the LCD while in boot section.

Press and hold the joystick ENTER(press down) while starting AVR Prog.



```
*
*    Purpose :         Cont
*
*************************
// mt __C_task void main
int main(void)
{
//    unsigned char stat
    unsigned char nextst
    // mt static char __
    PGM_P statetext;
    char (*pStateFunc)(c
    char input;
```

When AVR Prog… starts, release the joystick.



Hex file: Original program EEPROM.hex

Device: ATmega16 BOOT

Find the *.hex file you want to program with the “Browse” button, and press “Program”. See that “Erasing Device”, “Programming” and “Verifying” goes “OK”, this is done automatically. After upgrading the application, **press the “Exit”-button** in AVR Prog in order to leave programming mode in the ATmega169 boot loader.

Cycle the power (remove battery and power) and the startup display should now read “AVR BUTTERFLY GCC”

(remark mthomas: it is not necessary to cycle power, just move the joystick in the up-direction this lets the Butterfly leave the bootloader and start the application code but only if you’ve pressed the [Exit…] button in AVRprog before.)

Congratulations, you have just programmed the butterfly

Butterfly port assignments

Butterfly port assignments
ajb 11/12/2003
Yellow I don’t want to mess with
Green I can use

Port	Bit	Function		Connector	Connector
A	0	LCD	COM0	JTAG 8	
	1	LCD	COM1		
	2	LCD	COM2		
	3	LCD	COM3		
	4	LCD			
	5	LCD			
	6	LCD			
	7	LCD			
B	0		SS		PORT B 1
	1	Data Flash	SCK	ISP 3	PORT B 2
	2	Data Flash	MOSI	ISP 4	PORT B 3
	3	Data Flash	MISO	ISP 1	PORT B 4
	4	joy stick	OC0		PORT B 5
	5	Piezo	OC1A		PORT B 6
	6	joy stick	OC1B		PORT B 7
	7	joy stick	OC2		PORT B 8
C	0	LCD			
	1	LCD			
	2	LCD			
	3	LCD			
	4	LCD			
	5	LCD			
	6	LCD			
	7	LCD			
D	0	LCD			PORT D
	1	LCD			PORT D
	2	LCD			PORT D
	3	LCD			PORT D
	4	LCD			PORT D
	5	LCD			PORT D
	6	LCD			PORT D
	7	LCD			PORT D
E	0	AVR_RxD	RDX	UART 1	
	1	AVR_TxD	TXD	UART 2	
	2	joy stick	AIN0/XCK		
	3	joy stick	AIN1		
	4		SCL/USCK	USI 1	
	5		SDA/DI	USI 2	
	6		D0	USI 3	
	7		RST_FLASH		

F	0	Temp	ADC0	
	1	Volts	ADC1	Voltage Reader 1
	2	Light	ADC2	
	3	VCP	ADC3	
	4		ADC4	JTAG 1
	5		ADC4	JTAG 5
	6		ADC6	JTAG 3
	7		ADC7	JTAG 9
G	0	LCD		
	1	LCD		
	2	LCD		
	3	LCD		
	4	LCD		

(4/2004 Small update since a new pdf-file has been created with hopefully better quality)

Remark mthomas: I don't share Al's opinion about free and used pins for some pins mentioned in the above table. Here a copy of the text from the web-page http://www.siwawi.arubi.uni-kl.de/avr_projects/ about free or usable pins. Please take this as another opinion, as I don't know if I'm correct in all points.

> There have been some questions about "free" pins of the ATmega169 on the AVR Butterfly. Please refer to the schematics in the Butterfly user's guide. Most of the ATmega pins are blocked by the Butterfly on-board hardware and can not be used without loosing some functionality (esp. LCD).
>
> - The USI pins are free to use and available thru the USI connector.
> - The 4 JTAG-pins are also available if JTAG is not needed. Before the JTAG-pins can be used the JTAG-interface of the ATmega169V has to be disabled either by changing the JTAG-Enable-fuse via ISP of by programming the JTD bit in the MCUSCR register at application startup. So JTAG can be turned off with the JTD bit without an ISP connection.
> - If Chip-Select for the Dataflash is kept under control of the application the ISP-Pins might be used.
> - Taking the onboard level-shifter for RS232 into account the RX and TX Pins can be used.
> - Taking the onboard voltage-divider into account the Voltage-Reader Pin connected to the ATmega ADC-converter may be used
> - If setup and user-interaction in the application are separated or configuration is done via RS232 the pins connected to the joystick may be used (block physical access to the joystick).
>
> With some of these free I/O pins the number of inputs and outputs can be easily increased by using external shift-register integrated circuits like 74HC595 for outputs and 4021 (i.e. HEF4021B) or 74HC(T)165 for inputs .