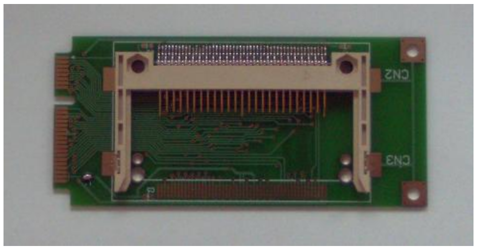
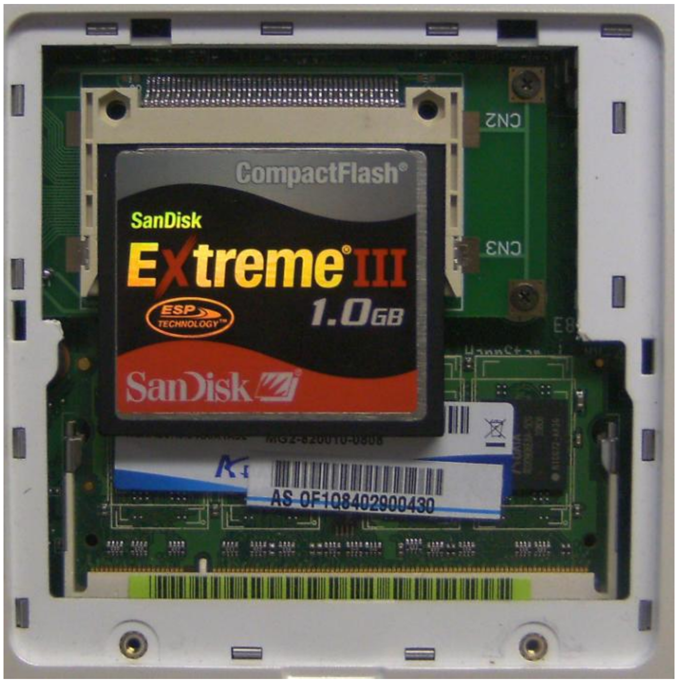
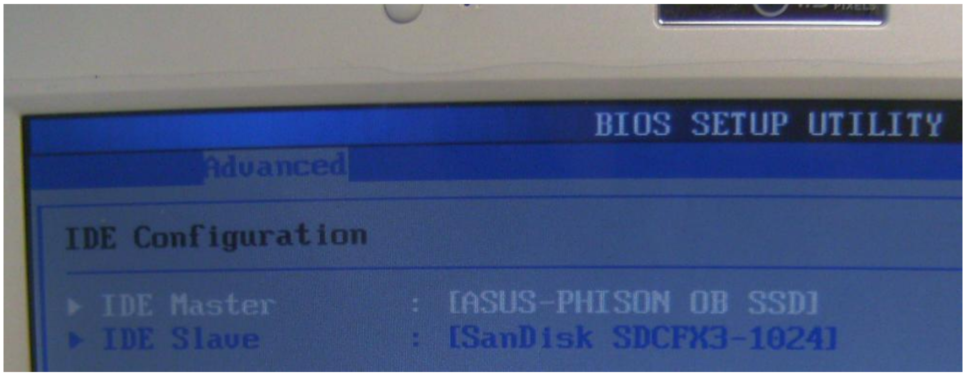
Antti-Brain

Issue 3

November 2008







```
                                     BIOS SETUP UTILITY
          Advanced

IDE Configuration

▶ IDE Master          : [ASUS-PHISON OB SSD]
▶ IDE Slave           : [SanDisk SDCFX3-1024]
```

Revised on November 30, 2008

Editorial

We had even less time as for the previous issue. December did come really quick this year.

So publishing November issue as is. With the hope to have somewhat more finished December issue.

Antti Lukats
Antti.Lukats@googlemail.com

http://groups.google.com/group/antti-brain

Cover Story

Place holder. Shots from some of my products.

Highlights

Various interesting findings.

Low power MCU

Seeking again low power MCU for one special use.

Device	VCC	@1MHz typ/max	Standby	Off/RAM	Price 100	Price 1000
MSP430F2001	1.8-3.6	220/270uA @2.2V	0.5uA	0.1uA	0.70	
ATtiny13A	1.8-5.5	190uA @1.8V	24uA			0.32
ATxmega	1.6-3.6					
MAXQ2000	1.8-2.7					3.26
MAXQ2010	2.7-3.6					3.45
EM6819	0.9-3.					

MSP430F20xx

- 1uS wakeup from standby
- Internal oscillators 16Mhz/32Khz
- 5 power save modes
- 1.2uA in active 4Khz/2.2V

Section incomplete in this issue ☹

MCU with Low Power LCD/RTC

Ok, here is a task, to find lowest priced MCU for battery powered operation with LCD and RTC capability.

Device	Mem	Price	Acceptable	Notes
ATmega169	16K	3.53 @100	No	Missing 32KHz oscillator
MSP430F412	4K	3.25 @100	Yes*	
MAXQ2000	64K	3.26 @1K	Yes	Mask option
ATxmega32A4	32K	3.08 @100	No	Missing LCD
PIC16C924	4K OTP	4.75 @100	No	KGD, no RTC
PIC16F913	7K	1.72 @5K	No	RTC?
PIC18F6363	8K	2.94 @5K	Yes	
PIC18F63J90	8K	2.35 @5K	Yes	
68HC908LV8	8K	?	No	No RTC, OBSOLETE also
MC9RS08LE4	4K	?		
MC9RS08LA8	8K	-		
MC9S08LL16	16K	1.74 @1000		RTC?
MC9S08LL08	8K	1.65 @1000		
MC9S08LC60	60K	?		
68HC908LJ12	12K	4.17 @100	Yes	

Actually I would not accept Microchip MCU, but it looked like best match at first comparison.

Short Stories

In this column we will list very short news, things that we found as new and interesting information in the time of reading.

Low profile LDO's

Are coming from Fairchild very soon, 0.6 and 0.5 mm package height versions.

New 3-Axis MEMS

ST, ADI, Freescale will all have new MEMS sensor announced in Q1 2009 (samples available now).

BT Low Energy

Several companies promise BT LE transceivers around 9/2009. That is 4-5 months after BT SIG issues the BT LE final specification. It is to be assumed that BT LE SoC's will not be available in 2009, but early 2010. Nordics nRF24LE1, looks like LOW ENERGY (LE) but it's baseband is not compatible to BT LE specification.

Altera MAX III

Is coming in 2009, but it will have other product name, specifications not know at present.

Spartan-4/Virtex-6

Xilinx says nothing. S-4 not coming 2008 for sure, and well Xilinx did not object when I said that if X-4 doesn't come 2009 it will be too late. As of some rumors Virtex-6 could be announced pretty soon, but that is a rumor only.

Actel 3x3mm

Actel has take the lead in smallest package for the FPGA. Well kind of.. those Actel devices that come in 3x3 package are not really FPGA's as they are more like rather small CPLD.

HS USB UART/FIFO Bridge from FTDI

FT2232H, check it out. We have not yet received samples or even confirmation on availability.

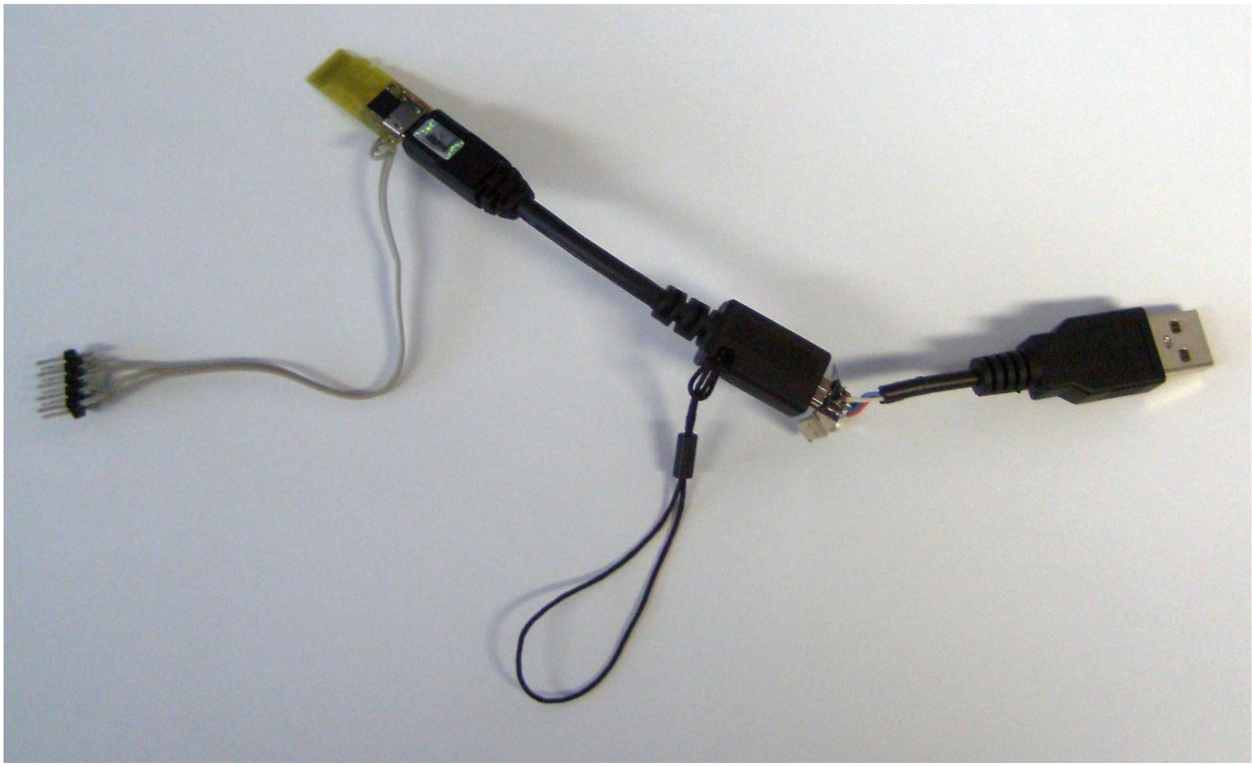
SiliconBlues

Swapped wires

While testing my first board with micro-USB connector I was really careful to measure and verify the connector footprint is not mirrored. Well it was! So I did think ok, I make special cable with wires swapped just for this prototype. I plug a micro-USB to mini-USB cable to my target board. And then I cut a mini-USB cable to make the wire swap (to fix the micro-USB mirror problem). I swap gnd-vbus and dm-dp wires. Plugging the cable, no smoke nothing hot, so far so good. Ok, loading some HID firmware into the MCU, Windows does see device attachment! But it is unknown device! Hum the gnd-vbus must be correct, could it be I actually did not swap dp-dm? Checking out, no they are swapped, so should be correct, just to be sure I swap them back and test, but again only unknown device is attached. On the prototype I had absolutely no bypass caps soldered, could this be the problem? Maybe the USB PLL doesn't work? I add the caps and test again, no difference. Ok let's look what is going on, I insert breakpoints into USB interrupt routine. USB Reset and Suspend are triggered but never any setup packet. The schematic is fairly simple barely any problem possible. So what is the problem? Have you guessed it? I didn't, did give up for a day. And almost instantly after that I realized the problem. Micro-USB connector has 5 terminals, with 4 wires connected to it. So no matter I swapped the wires in the cable, I had never both dm and dp connected! Only dp (pin 3) had actual wire connected, so my swapping connected it to either dp or dm. As dp has pullup for FS devices so both swap variants did give the USB detect for the host, but dm was never connected so also not data transfer was ever seen.

The next day.

I add wire pin 2 to 4 on micro-USB pins on my board (so that the missing pin connection should be available). I try, not recognized properly. Hm.. maybe I can modify the cable still I cut the mini-USB cable very close at the connector, eh it looks bad, all inside the plastic mold. By looking at, I managed to cut one data wire so that caused some next headache. When I found that, I did go digging deeper and removed all the plastic, and soldered directly to the connector. Again tried to be very careful, and tested many times not to mess around with vbus-gnd swap. Trying out, device attach is seen by windows, but eh after it sees the device once, the USB hub port stops working? I disconnect dp/dm, still it sees attach? What? I plug in again, and see small smoke. Power off. Testing again, and well it was vbus-gnd swapped! Fixing again. And big relief the IC is still working. And it is even recognized as USB device!



Working setup: micro-USB to mini-USB cable (original), and “swap” mini-USB cable.

Controller Corner

SiLabs MCU's Part I

NOTE: the first part is unfortunately not the getting started part, some topics are probably for more advanced user.

First of all SiLabs MCU's are the Cygnal ones and those are just another 8051. Just another 8051? Well the SiLabs MCU's have some nice features. SiLabs was one of the first to offer crystal less FS USB device operations.

Getting Started

While I did invent SimmStick™ so did SiLabs introduce ToolStick. Later many other companies have started to offer low cost development tools in similar fashion.

Now while there are tons of simple DIY programmers and software for Microchip PIC's and Atmel AVR's there next to nothing for SiLabs MCU's. There is one project for Linux using simple parallel port hardware, except that all tools are commercial. There is also project in progress to support SiLabs C2 programming interface for Nintendo DS handheld game console. But the fact remains, if you have a PC, then even if it has old style LPT port, you cannot just connect a few wires and hope to get your SiLabs MCU programmed. Well this is true for all newer MCU's with C2 ISP/Debug interface, the older devices have JTAG interface for debug/ISP and SiLabs offers a free SVF generation tool.

There are different ways to get started:

Choice	$	What you get	Time spent (till first success)	Equipment trashed	Learned
DIE-HARD	0*	Bad looking adapter and/or proto board	>= 1 day to forever	1 Desktop PC	From nothing to guru level
Lucky	0	What you asked ☺	<= 1 day	nothing	
ToolStick-EK	9.90	C2 Adapter, F300 target	<= 1 day	ToolStick-EK	
ToolStick-Starter	24.90	Debug Adapter (C2/JTAG,UART!), F330 target	<= 1 day		
ToolStick-BA, ToolStick-Debug, ToolStick Capsense Target	17.90+ 8.90+ 19.90 ==== **46.70**	Debug Adapter (10 Pin Header, C2, JTAG, UART), F931 with CapSense	<= 1 day		

Investment 0.00 USD Die-HARD way

You order free samples of some JTAG device 0xx, 1xx, 2xx series and from some newer (C2 interface) device also. Beware that sometimes the customs cost may still apply, it depend the country you live in, and the CUSTOMS VALUE in the shipping documents. The free samples are sent by Mouser and while the samples and shipping are free, Mouser adds a customs invoice with the catalog pricing of the samples. This may trigger the customs clearance issues.

JTAG Device: You use some of your existing JTAG cable/programmer and SVF files generated by the SiLabs utility. Or you make up some simple LPT JTAG cable (wires + resistors if you are lucky).

C2 Device: You build a simple LPT interface for C2, and use the Linux Software to program your device.

Now while the above is possible, please do not take it as recommended path. It's doable, and some Die hard DIY people would succeed. But it not worth the hassle money-wise.

Investment 0.00 USD easy way

Grab a free trade show ticket, go to the show, talk to SiLabs at their booth, chances are you get something for free – either a ToolStick kit or full Development kit. Or ask your local distributor if you have one in your country. They may also give you a free ToolStick. (Has happened at least in the past).

Should you get it, DO NOT GIVE IT AWAY. I had two Development kits, and a ToolStick-EK what I donated to those in more need then I, but when I needed a C2 Programmer, I had no working solution (the last programmer did burn). And cheapest option to get one the next day would have cost me around 180 USD as SiLabs has no direct distributor at my current location, and the catalog shop had only expensive options orderable at high cost.

While at SiLabs booth, try get full development kit, or ToolStick-Capsense kit if they have it ready. If not take whatever you get ☺

Investment 9.90 USD (+S/H)

You buy the ToolStick-EK (9.90$ list price) and cut it in two halves. You get a C2 programmer and C8051F300 target board. Of course you can develop and experiment with first before using the saw.

Warning: if you reconnect the F300 halve to the programmer halve and reverse the supply, then the C8051F321 in the programmer part with burn. The LDO goes pasta, and chances are that after adding external 3.3V LDO the programmer part is still unusable. The F300 remains working.

Hum, you could be eligible to add free samples to the ToolStick-EK order. So you would get a development tool, one target board and samples for total 9.90 USD.

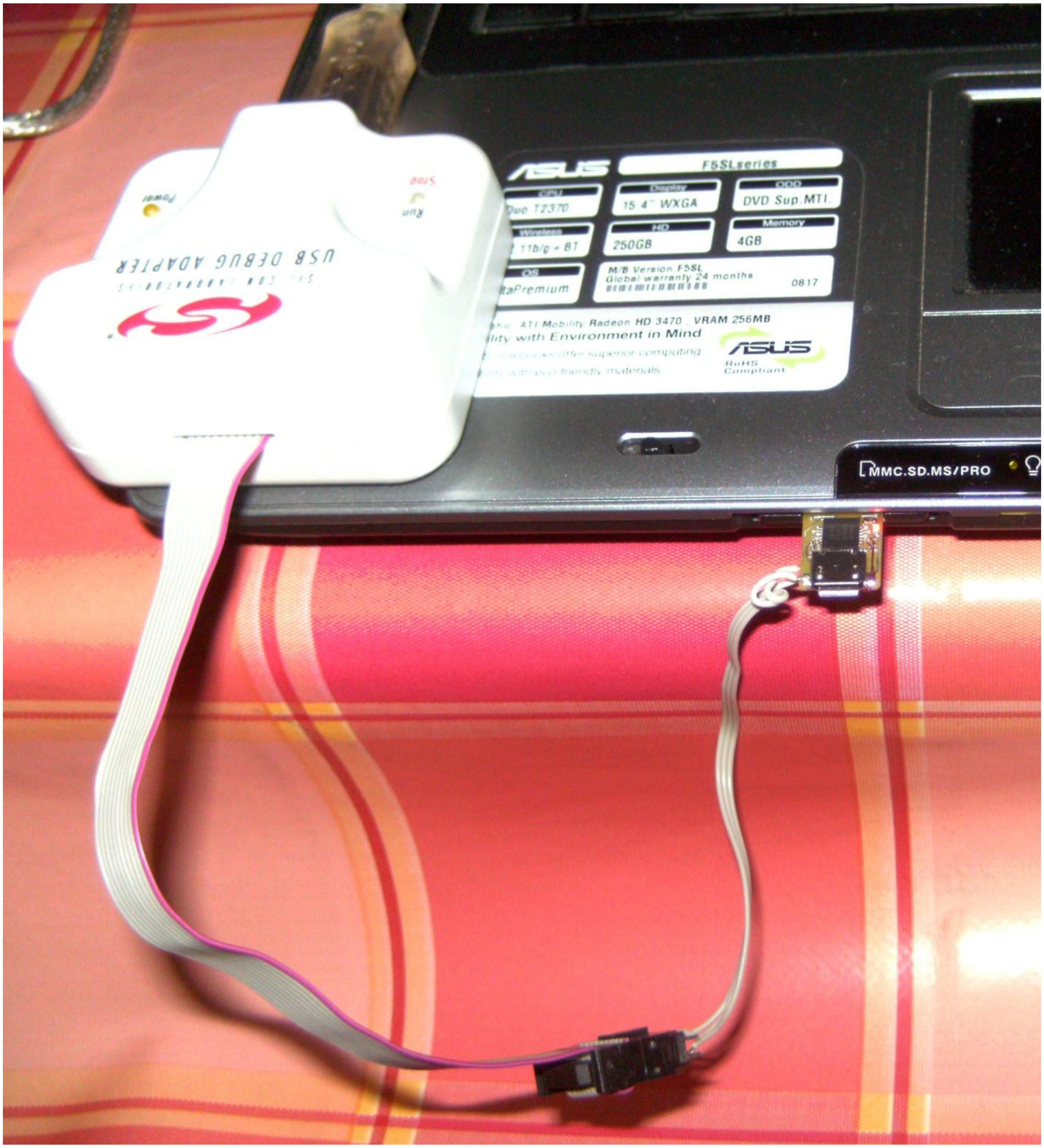
This solution will offer full integration with SiLabs IDE and Debug solution for all devices with C2 interface.

Investment 24.90 USD (+S/H)

You buy the ToolStick starter kit. Recommended.

Using SDCC

I usually never consider using SDCC, but troubleshooting of the licenses is real PITA sometimes, so when someone asked ”have you considered SDCC?” – I decided to give it a try. Downloaded latest snapshot, only 2MByte, installed it. Changed toolchain integration to SDCC in SiLabs IDE. And now interesting, does anything compile? Blink project seems to compile fine. Connect, Download, Run, and yes the LED blinks. So the SDCC compiler integration with SiLabs IDE works at least for some projects out of the box.

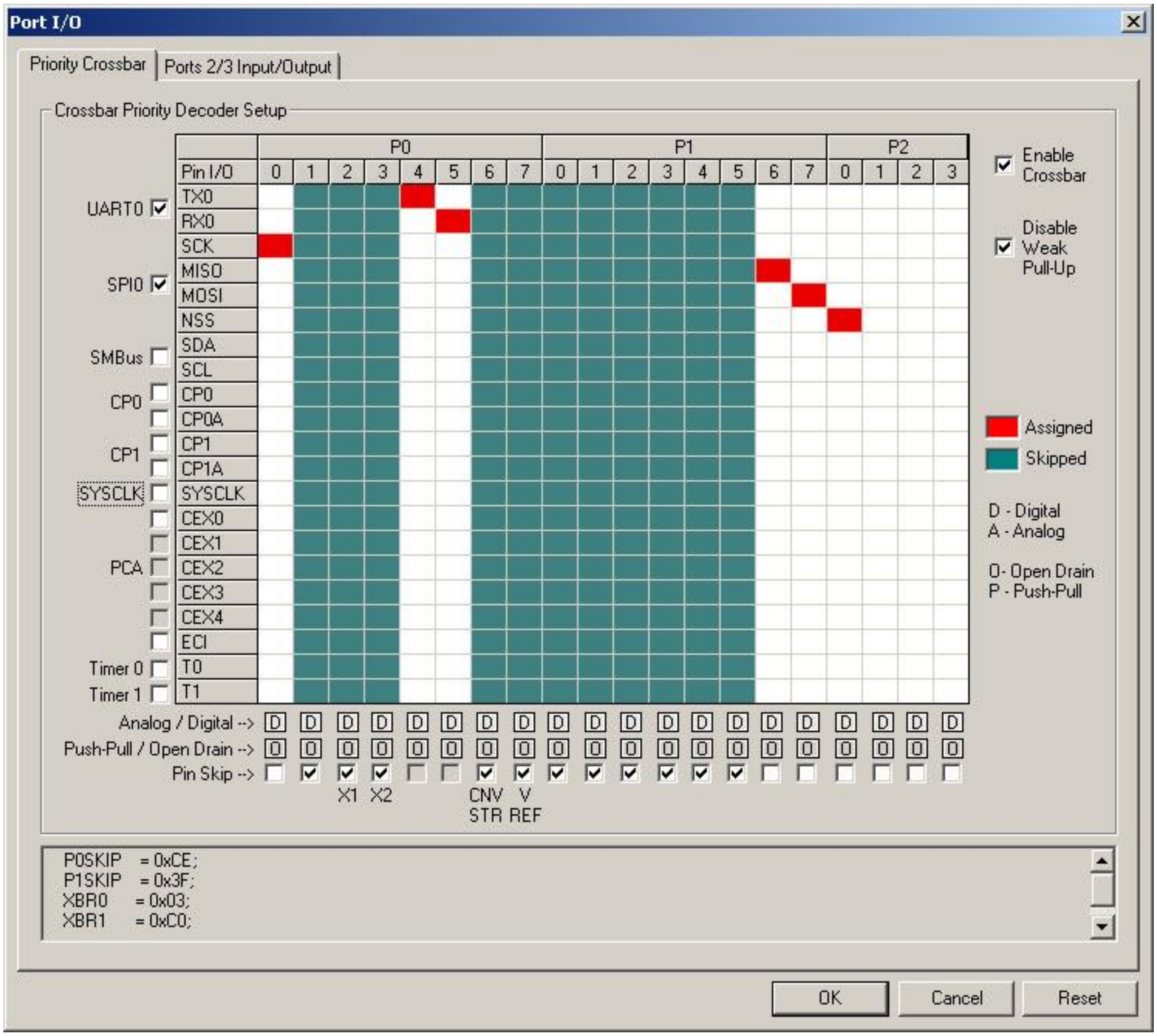


The LED is actually blinking – I did multiple shots until I got one where it is lit. The target device is C8051F321 on micro-UD card that is inserted into the notebooks SD slot using micro-SD to SD adapter. This screenshot is take from first prototype of the micro-UD USB adapter card during initial Bootloader development. Well while a LED Blink project did compile with SDCC, all USB projects failed with some compiler errors. Maybe they are easy to fix in source code, but I had no time for that so I reverted back to Keil C compiler.

Ah, on SiLabs web are User Forums, and there a special section dedicated to SDCC. So at least some hints are available to aid the conversion.

Crossbar

This is a special feature a priority decode – so called crossbar, it allows some reassignment of the peripherals. But it also heavily limited so playing with the crossbar should be done at early stage of the development.



		P0								P1								P2			
	Pin I/O	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3
UART0 ☑	TX0		🟩	🟩	🟩	🟥		🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
	RX0		🟩	🟩	🟩		🟥	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
SPI0 ☑	SCK	🟥	🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
	MISO		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟥					
	MOSI		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩		🟥				
	NSS		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩			🟥			
SMBus ☐	SDA		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
	SCL		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
CP0 ☐	CP0		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
☐	CP0A		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
CP1 ☐	CP1		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
☐	CP1A		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
SYSCLK ☐	SYSCLK		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
☐	CEX0		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
☐	CEX1		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
PCA ☐	CEX2		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
☐	CEX3		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
☐	CEX4		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
☐	ECI		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
Timer 0 ☐	T0		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
Timer 1 ☐	T1		🟩	🟩	🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩						
Analog / Digital -->		D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Push-Pull / Open Drain -->		O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O
Pin Skip -->		☐	☑	☑	☑	☐	☐	☑	☑	☑	☑	☑	☑	☑	☑	☐	☐	☐	☐	☐	☐
				X1	X2			CNV STR	V REF												

☑ Enable Crossbar

☑ Disable Weak Pull-Up

🟥 Assigned
🟩 Skipped

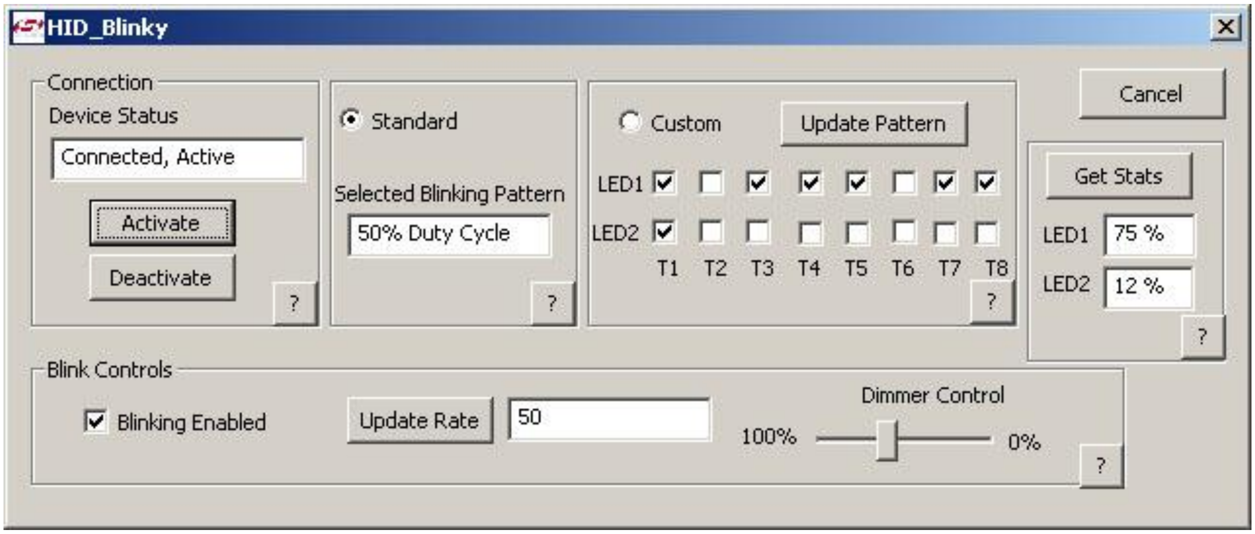
D - Digital
A - Analog

O- Open Drain
P - Push-Pull

```
P0SKIP   = 0xCE;
P1SKIP   = 0x3F;
XBR0     = 0x03;
XBR1     = 0xC0;
```

Here is my work project where I tried out to save pins, the design required a 8 bit parallel port, where upper bits 6 and 7 can optionally be SPI MISO/MOSI. At the beginning I did think I need use 2 I/O pins each to accomplish the task but little playing with the SiLabs Configurator2 did show that I can actually move the SPI pins to the locations that are needed. I reserved first the UART pins (they cannot be moved). After that I moved the MISO/MISO to their needed locations. The result is shown in the screenshot. I could move the SPI SCK to any free pin in P0, but I cannot use any other hardware functions in the skipped P0 pins. Only 3 pins in P2 are still free to be assigned to some hardware functions (I^2C/SMBus, SYSCLK or some other). Now how to decide about the rest of the pins? Well after reading the datasheet I figured out that interrupts can be assigned to any P0 pins. So signals that may be used as GPIO interrupts should be assigned to the free pins in P0.

Another test, small board with C8051F321, trying SiLabs "HID Blinky" demo. The LED doesn't blink. Well in the original they had LED's on P2.2 and P2.3, my board had only one LED on P2.0, I had modified GPIO setup (to make P2.0 output) but it dint seem to be enough. I tried some debugging all seemed Ok. The reason was crossbar setting, I had to change it, the LED blink was done by the PCA (Programmable Counter Array). So the crossbar had to be modified to connect timer output to P2.0, after that I could use the ready built GUI to control the LED blinking and intensity.



Device Status: Connected, Active

● Standard

Selected Blinking Pattern: 50% Duty Cycle

○ Custom

	T1	T2	T3	T4	T5	T6	T7	T8
LED1	☑	□	☑	☑	☑	□	☑	☑
LED2	☑	□	□	□	□	□	□	□

LED1 75 %

LED2 12 %

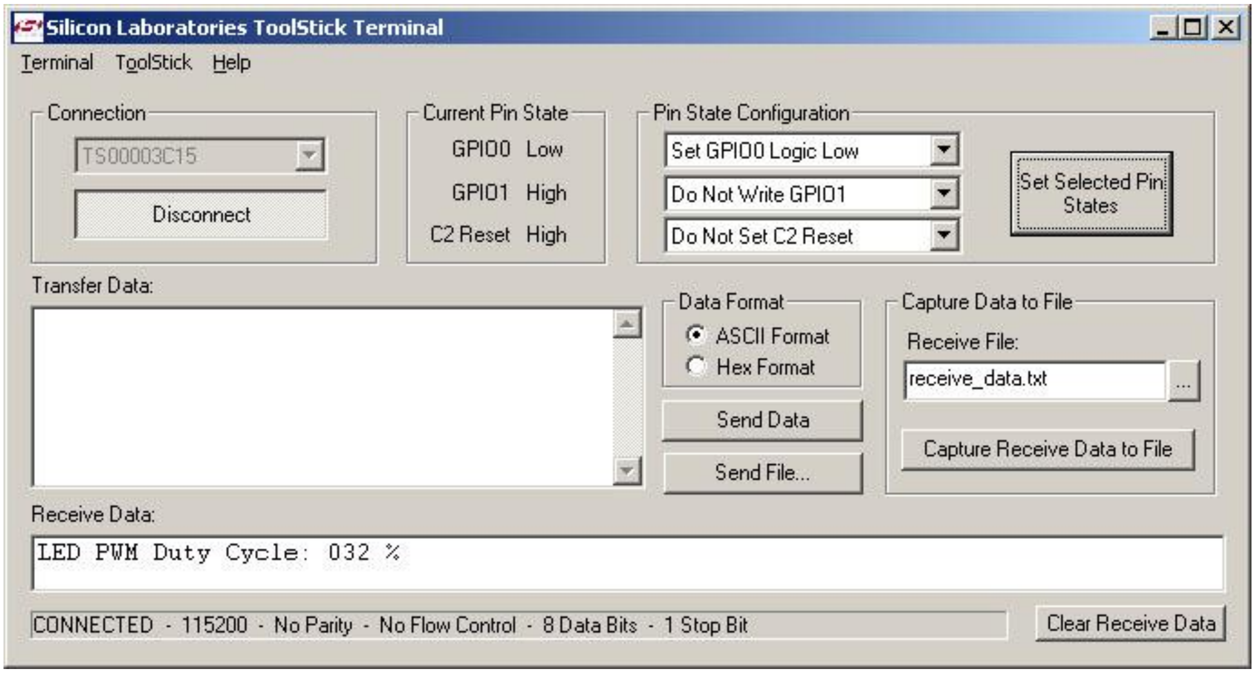
☑ Blinking Enabled

Update Rate 50

Dimmer Control 100% 0%

ToolStick-BA+DC

This is the best way to get started, you either get the ToolStick starter package (BA+DC330) or you get the ToolStick-BA and DC or your choice. From Electronica 2008 I had got one Starter package so I tried it out as well. Plugged it into the USB connector, fired up the IDE, select Examples/ToolStick/DC330/Features Demo, Build Project, connect, download, run. And the LED blinks, and the potentiometer can be used to change the blink rate as well. Now there is also a ToolStick terminal program (including source code) provided by SiLabs. So I start it. But it did not show any devices to be available. Ok back to SiLabs IDE, disconnect. Close terminal, Start terminal. And now it works.



Connection: TS00003C15

Current Pin State
GPIO0 Low
GPIO1 High
C2 Reset High

Pin State Configuration
Set GPIO0 Logic Low
Do Not Write GPIO1
Do Not Set C2 Reset

Transfer Data:

Data Format
● ASCII Format
○ Hex Format

Capture Data to File
Receive File: receive_data.txt

Receive Data:

```
LED PWM Duty Cycle: 032 %
```

CONNECTED - 115200 - No Parity - No Flow Control - 8 Data Bits - 1 Stop Bit

GPIO level is used to set the behavior from blink to PWM, then the status is sent back from F330 to the terminal when the potentiometer is adjusted.

Packages

		Size	Pitch			Devices (F is omitted)
	TQFP100	16x16	0.5			020,022,040,042,044,046,060,062, 064,066,120,122,124,126,130,132
	TQFP64	12x12	0.5			
	TQFP48	9x9	0.5			348
	QFN48	7x7	0.5			
	LQFP32	9x9	0.8			310,312,320,314,349
	QFN32	5x5	0.5			930-GM
	QFN28	5x5	0.5			311,313,315,321,326,327
	QFN24	4x4	0.5			316,317,338,339,921-GM,931-GM
	DIP20		2.54			330-GP
	TSSOP20	6.5x6.5	0.5			
	QFN20	4x4	0.5			330-GM,336,337
	SOIC14		1.27			T600,T601,T602,T603,T604,T605
	QFN11	3x3	0.5			300,301,302,303,304,305
	QFN10	3x3	0.5			520A,521A,523A,524A526A,527A

Easy hand solder packages are marked green. There is surprisingly one TH package DIP20 available. What is too bad is that the SOIC packages are only available for OTP versions. SiLabs says the OTP devices are pin-compatible to Flash version, but other way around is not true, there are no Flash devices in SOIC. Soldering the 0.5mm pitch TQFP's isn't so hard either, well SiLabs has even an application about the soldering of them AN114. The QFN packages are little bit harder to solder without hot-air soldering station. If the PCB is done with IPC dense then soldering with solder iron is almost impossible, I have tried and had real hard times. If the QFN pads are little longer outside the package on the PCB then QFN's can also be soldered with solder iron.

IdeaREG™

IdeaREG online service is not yet open, but the following should be considered as submitted and registered idea(s).

Special Offers

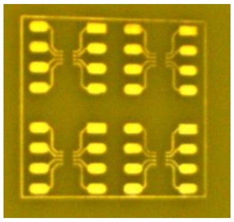
XMOS XC1 Promotion

https://www.xmos.com/xc1_promo_details

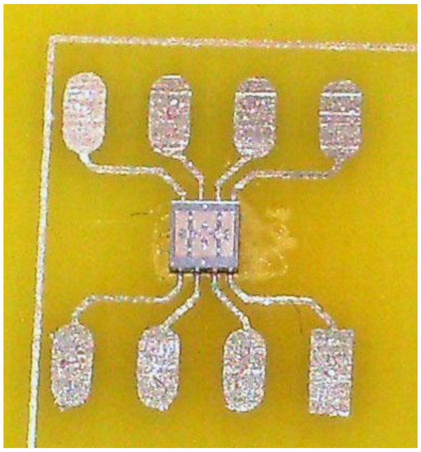
Single Sided

BGA-8 Adapter

I have done some press-and-peel PCB before; now when having office at local PCB manufacturer, I wanted to give it another try in professional environment. They have a GBC laminator, so I used it for the heat transfer. The very first try was not satisfactory, either too low temperature or too low pressure or both. Results of second attempt are here:



This is just for test a break-out-board board for a specialty 8 ball BGA package. The trace are less than 6 mil's at the balls. I had placed 4 patterns in the hope one would be Ok, but all 4 are close to perfect.



This is the same board with the IC soldered. The IC is 2 by 2 mm large (small?). Do not look Pin 1 marking be weird edge, I was too lazy to mirror as the design is otherwise symmetrical. Actually I forgot the Pin 1 marking makes it not fully symmetrical.

If you look at this page please zoom out to 12.5% it shows the board with the IC in about real size ☺

0-Security

I have a unfinished book with preliminary title “0-security”, as long as it remains unfinished and unpublished I will publish pieces of new security related information here.

DSi Exploit (string overflow)

DSi (Nintendo **D**ual **S**creen with **i**nternet) was released 1 Nov 2008 in Japan only. Within limited hours (maybe a day?) it was reported as hacked already. How come? Well it is to be assumed that the first reported hack was already prepared by an Japanese hacker, so he sent the hack to his friend who had received a DSi and could test it. And it was success – that is the hack worked.

DSi has DS backward compatible mode and supports “old” DS ROM game titles. There are about 2900 of DS titles released as of today. Those ROMs are real ROM’s that is they are factory mask ROM version. No way to change the code. Also there are two different encryption methods used for the ROM interface. Well while those are hacked also, the DSi hack was not doing anything with the ROM code, but used errors made by the game publishers. DS ROM’s include small EEPROM serial memories to save user preferences and game save data. While those EEPROM’s are accessed by the low level API’s from Nintendo SDK (and those are most likely tested well), the use of the data is done by the software of the game publishers. Now the hackers did think correctly that it is not possible that the save data processing routines do not have missing “overflow” tests. What makes me wonder is that within a few days 5 different ROM titles was found with such exploit possibility.

So the “exploit toolkit” modifies the save data to have some string to be too long. It also fixes all checksum in the data areas. When now the DS title save EEPROM is programmed with this exploit data, then the ROM code will have internal stack overflow what causes the execution to jump to code loaded from the save data and resident in RAM.

References

- << >>