Flashburn

**Download code**

Before downloading the code a thorough debugging must be finished Once the code is in the flash debugging is excluded. When downloading code to the flash some adjustements must be done. An assembly file must be added. The linking file \*.cmd and the interrupt vector must be changed  Furthermore has the executable \*.out-file to be converted to a hex-file. The hex-filen can the be downloaded to the flash memory and executed. For more information about flash burning, see [Texas Instrument  documents.[PDF](http://www.hh.se/download/18.70cf2e49129168da0158000148376/1216811264161/spra804.pdf)](http://www.hh.se/download/18.70cf2e49129168da0158000148376/1216811264161/spra804.pdf) (pdf, 169.1 kB)

**Adding files to the project**

Add a lib-file to the project: C:\ti\c6000\cgtools\lib\rts6201.lib. A [fboot.asm[ZIP](http://www.hh.se/download/18.70cf2e49129168da0158000145464/1216731408168/fboot.zip)](http://www.hh.se/download/18.70cf2e49129168da0158000145464/1216731408168/fboot.zip) (zip, 1.1 kB) will then be added. Change the first lines in the interrupt vector to:

.ref \_timer0\_isr  
.sect "vectors"  
.ref \_boot

RESET\_RST:  
mvkl .S2 \_boot, B0  
mvkh .S2 \_boot, B0  
B .S2 B0

which makes the PC initially jumping to  \_boot instead of \_c\_int00.

Chenge in yout own linker file  \*.cmd as

MEMORY  
{  
INT\_VEC       o = 00000000h  l = 00000200h  
BOOT\_RAM:  o = 00000200h  l = 00000200h  
IRAM:           o = 00000800h  l = 0000f7ffh  
}

SECTIONS  
{  
"vectors"      > INT\_VEC       fill = 0  
.boot\_load    > BOOT\_RAM  fill = 0  
.text            > IRAM             fill = 0

and compile as ussually. Remark: This out-file is not suitable for debugging puposes. This since it mixes the download file with the file its own file.

**Convertering \*.out to \*.hex**

To control the converter program acommand-file. Copy this and edit it: Change  ex1.out to what your out file is named. A map-file is then created [out2hex.map[ZIP](http://www.hh.se/download/18.70cf2e49129168da0158000145460/1216731409865/out2hex.zip)](http://www.hh.se/download/18.70cf2e49129168da0158000145460/1216731409865/out2hex.zip) (zip, 306 bytes) . On line

FLASH: org = 000h, len = 0x10000,romwidth = 8, files = {ex1.hex}

can  ex1.hex be change to your name.

**Configure flashburner**

Copy [the files[ZIP](http://www.hh.se/download/18.70cf2e49129168da0158000145463/1216731408666/fbtc.zip)](http://www.hh.se/download/18.70cf2e49129168da0158000145463/1216731408666/fbtc.zip) (zip, 5.3 kB) to your directory and start Codecomposern, choose Tools and Flashburn. Choose file and Open in the flashburner program, and choose the file FBConfig.cdd and then Ok.  In the top filed has your command-file to be given. If the file not is in the same directory has the complete path to be given. Be careful in naming and downloading your files. It is easy choosing the wrong one. In the second field has the hex-file to be added. Choose processor and  FTBC.out file. The burn program will use this FTBC.out-filen to communicate with the DSP. Choose memory adress 0x90000000 and size 0x10000. Then the text Not Connected has to be altered to Connected. If this isn't the case, press save and close the document and then reopen it.

**Download the program**

Now will the chain-icon be in one piece and the text connected be visible. To create the  hex-filen you press Start Programming.  But if is the first time the system will look for a hex-file which does not excist. (Most probably a program bug). So, in this case you can choose any excisting hex file. i.e. [FlashBlink.hex[ZIP](http://www.hh.se/download/18.70cf2e49129168da0158000148725/1216811472207/flash-blink.zip)](http://www.hh.se/download/18.70cf2e49129168da0158000148725/1216811472207/flash-blink.zip) (zip, 2.4 kB) then you have to replace it with your own hex file. Press Erase  folowed by Start Programming This will take approximately 100 sec. Close CodeComposer and unplug the power supply to the DSP. When the DSP is restarted, your code will be copied from flash to the internal RAM and will the start its execution from there.

**How does the  program work at reset**

At reset is a 1 kb block copied, using a DMA.channel, from DSP flash to address 0 in the internal RAM. Meanwhile is the Program Counter stalled in idle. Once the DMAtransport is finished the program starts at address 0. The program starts in what is known as the Reset vector. From this address it branches to assembly routine which cpoys the rest of the program in to RAM. Finally the PC is adjsusted an the downloaded program runs.