

How to copy the flash contents of an unsecure device and reprogram it on another device

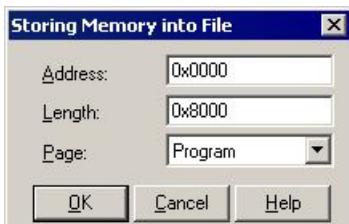
Follow this procedure if a simple save of the flash contents (as a COFF i.e. a xxxx.out file) and reprogramming it using the CCS flash plugin does not work.

You need to dump the flash contents to a text file, and then create a new CCS project that creates a new .out file from the data. Here's what you'd need to do:

- 1) In CCS, do File->Data->Save. Name the filename to save to, and choose "Hex" as the Save as type. Click "Save".



You will be prompted for memory to save. Save the entire flash contents. For LF2407A or LF2406A for example, this is 0x0000 to 0x8000. Make sure to choose "Program" for the Page. Click OK.

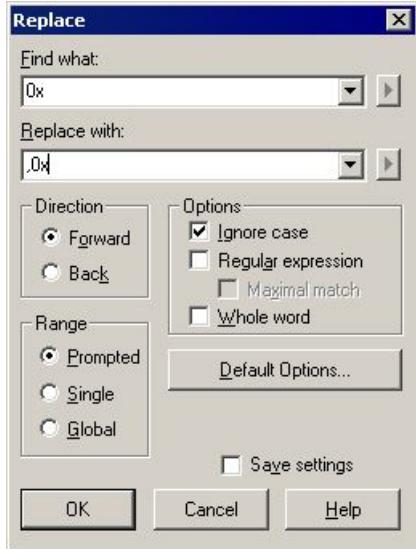


- 2) Rename the saved file with a .c extension instead of the .dat extension.
- 3) Start CCS, and create a new CCS project. Add the .c file to it.
- 4) Use the CCS editor to edit the .c file. It will look like this:

```
1651 1 0 0 8000
0x7980
0x0044
0x7980
0x0002
0x7980
0x0004
0x7980
0x010B
0x7980
0x0000
```

You need to turn the data into a constant array in C code. Do this:

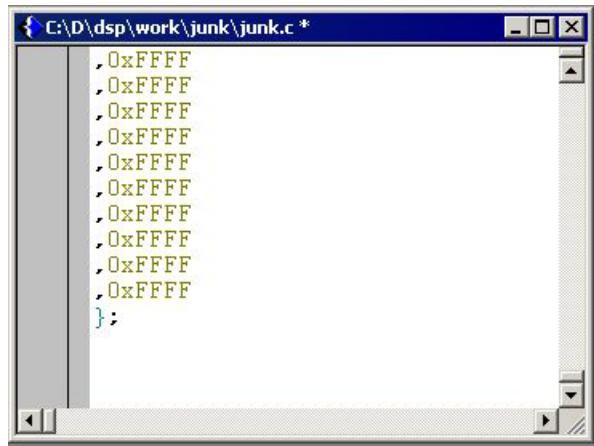
- Delete the entire first line. That is, "1651 1 0 0 8000". This line is not part of the flash contents.
- Insert a new first line that is this "const int x[32768]={"
- Use find/replace in the editor to replace all occurrences of "0x" with ",0x". That's because array values need to be separated by commas in C.



- Delete the comma in front of the very first value:

```
const int x[32768]={  
    0x7980,  
    ,0x0044,  
    ,0x7980,  
    ,0x0002,  
    ,0x7980,  
    ,0x0004,  
    ,0x7980,  
    ,0x010B,  
    ,0x7980,  
    ,0x0008,  
    ,0x7980,  
    ,0x000A,  
    ,0x7980}
```

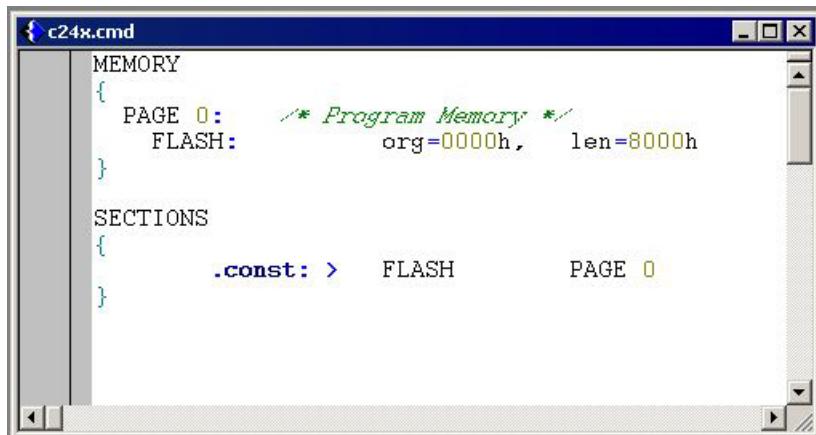
- Make sure you terminate the array properly with a closing brace and semi-colon. Also insert a carriage return at the end since CCS wants source files to end with a carriage return (it will complain otherwise).



```
C:\D\ dsp\work\junk\junk.c *
char const test[] = {0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF};
```

f) Save and close the .c file.

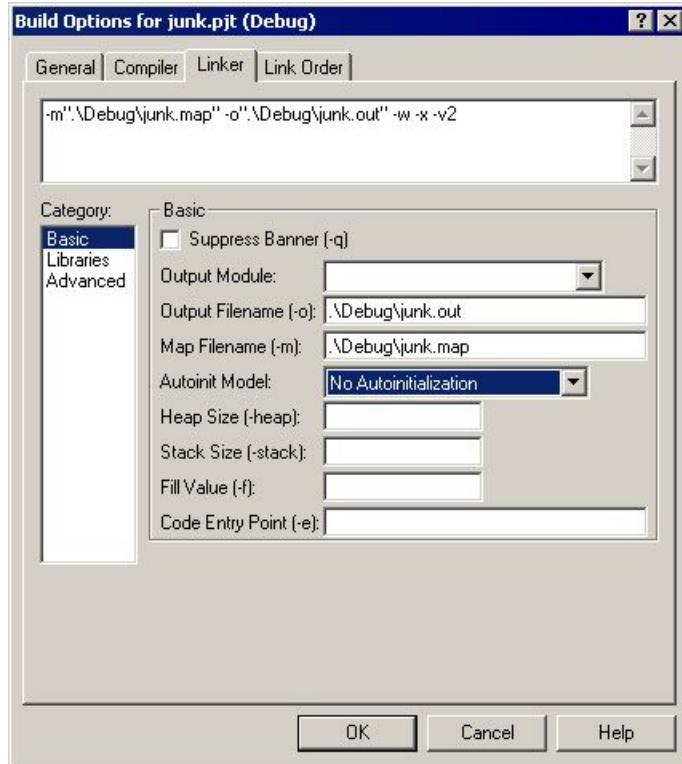
5) Create a new linker .cmd file that defines the flash memory in your device, and links the .const section there. Like this:



```
c24x.cmd
MEMORY
{
    PAGE 0: /* Program Memory */
        FLASH:          org=0000h, len=8000h
}
SECTIONS
{
    .const: > FLASH      PAGE 0
}
```

Add the .cmd to your project.

6) In CCS, go to Project->Build_Options, and choose Linker tab. Change the autoinit mode to "No Autoinitilization". This keeps the _cint00 routine from the rts library from getting linked in.



7) Build the project. It should build without error. You may get a warning about -pd and -ps options being ignored. That is OK. That is a bug in CCS, and can be ignored.

8) You can inspect the .map file to see that the build worked correctly. You should see the flash completely populated with the .const section:

output section	page	origin	length	attributes/ input sections
.const	0	00000000 00000000	00008000 00008000	junk.obj (.const)
.text	1	00000000 00000000	00000000 00000000	UNINITIALIZED junk.obj (.text)
.data	1	00000000 00000000	00000000 00000000	UNINITIALIZED junk.obj (.data)
" data-bbox="118 558 875 855"/>

```
*****  
TMS320C24xx COFF Linker      Version 7.04  
*****  
>> Linked Mon Jan 05 10:13:15 2009  
  
OUTPUT FILE NAME:  <./Debug/junk.out>  
ENTRY POINT SYMBOL: 0  
  
SECTION ALLOCATION MAP  
  
output section  page    origin        length      attributes/  
              -----  -----  -----  
              .const    0      00000000    00008000      input sections  
                      00000000    00008000      junk.obj (.const)  
  
              .text     1      00000000    00000000      UNINITIALIZED  
                      00000000    00000000      junk.obj (.text)  
  
              .data     1      00000000    00000000      UNINITIALIZED  
                      00000000    00000000      junk.obj (.data)
```

9) You can now use the flash utility to program the .out file you just created

