TI Developer Conference
February 18-20, 2004 • Houston, TX • Westin Galleria Hotel
Connecting Real People with Real Solutions

TMS320F281x™ Flash Programming Solutions

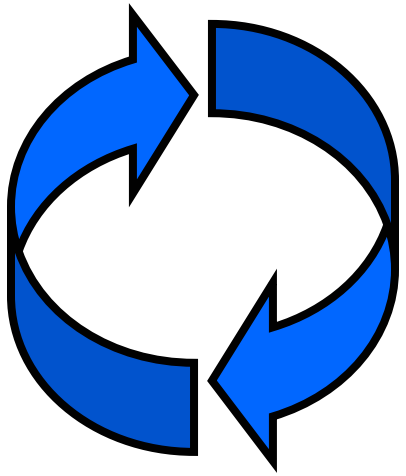
Texas Instruments

Flash Programming Solutions

When should I think about Flash programming solutions?

Flash programming can occur in all phases of a product's development cycle:

- Firmware debug
- Prototype units
- Production programming
- Field updates



Agenda

- **Section 1: Flash 101: Understand F281x Flash programming fundamentals.**
- **Section 2: Learn about programming solutions for the development and prototype phase:**
  - **Code Composer Studio Plug-In**
  - **SDFlash from Spectrum Digital**
- **Section 3: Understand how you can develop for custom solutions, field updates and production programming.**
  - **Flash programming API**
  - **Embedding Flash programming solutions**
  - **Custom programming solutions**

Section 1

Flash 101

Understand F281x Flash Programming Fundamentals

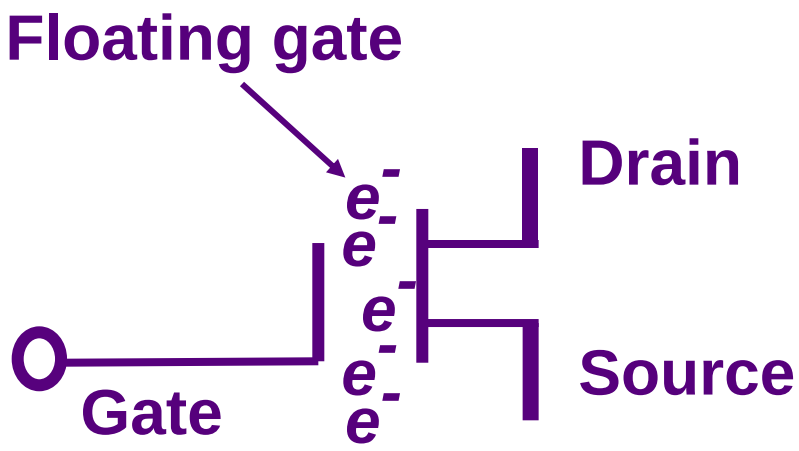
How to prepare the hardware for Flash programming.

What operations are needed to program the Flash?

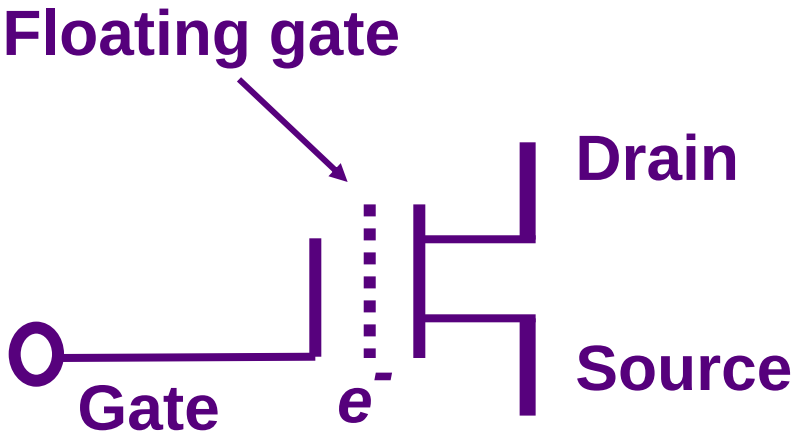
What does it mean to erase/program Flash?

Flash Cell Structure

What does a Flash cell look like?



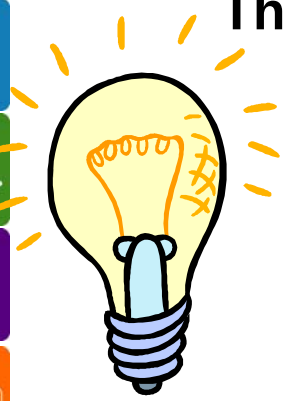
A cell with charge on the floating gate: value of 0



A cell with little charge on the floating gate: value of 1

CPU Based Programming

F281x devices are programmed via time-critical algorithms that execute on the DSP



The algorithms include timing-critical delay loops:

- Algorithms must be executed from single-cycle SARAM.
- You must configure the algorithms for the CPU frequency.
- To insure proper verification, execute the algorithms at the fastest CPU frequency for your system.
- The algorithms should not be interrupted!

Voltage supply:

- A 3.3 V supply must be applied to the VDD3VFL Flash voltage supply pin.

  Note: this voltage is used for programming AND reading the Flash. Thus it should always be connected.

Algorithm Operations

What operations are needed to program the Flash?

- **Erase:**
  The erase algorithm gradually removes charge until all of the bits within a Flash sector read back 1's.

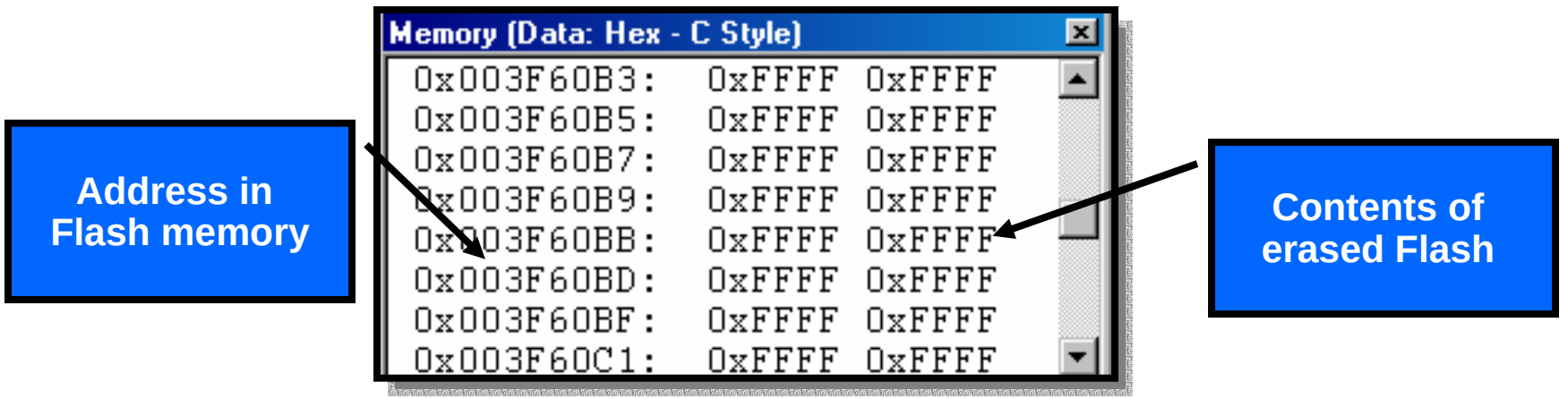
  The erase algorithm consists of 3 steps:
  1. **Clear: Program all the bits in the sector to 0.**
  2. **Erase: Sets all the bits in the sector to 1's.**
  3. **Compaction: Corrects any "over-erased" (depleted) bits.**

- **Program:**
  Program puts your application code and/or data into Flash by gradually depositing charge on specified bits until they read back 0.

The Erase Operation

Erase FAQ's

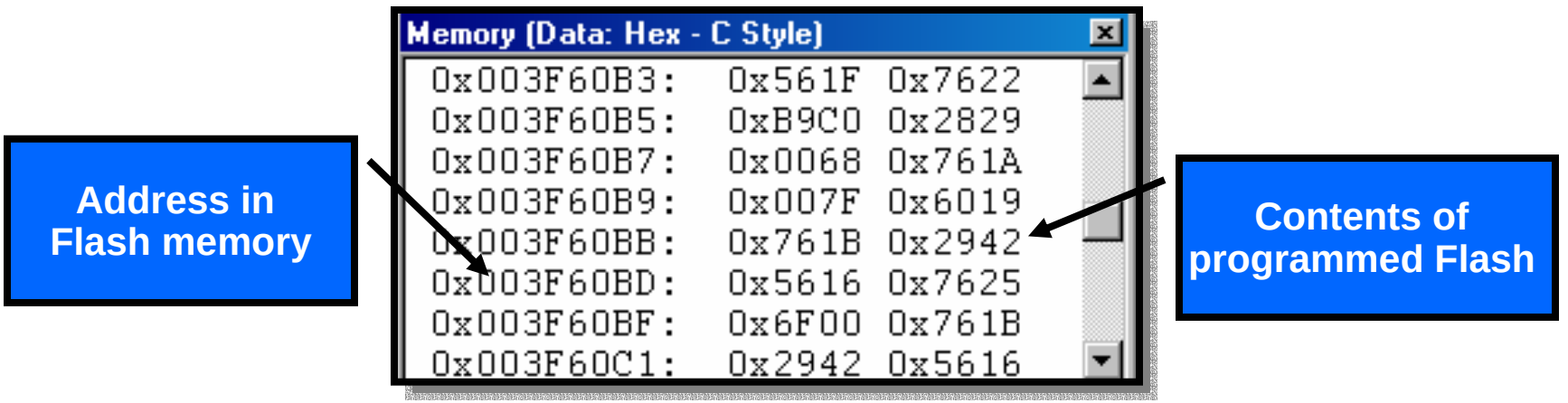
- ✓ Flash comes from the factory in an erased state.
- ✓ The erase algorithm sets <u>all the bits</u> in a sector to 1.
- ✓ The minimum amount of memory that can be erased at a time is a sector.
- ✓ Erase operates on Flash only. OTP cannot be erased.



The Program Operation

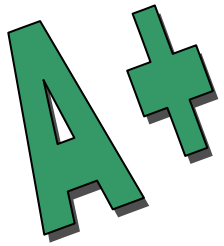
Program FAQ's

- ✓ Program is used to set bits within the Flash to 0.
- ✓ Program CANNOT move a bit from a 0 to a 1.
- ✓ Program operates on both Flash and OTP.
- ✓ Program operates on single bits with a 16-bit block.



Review of Flash Basics

Flash 101 Quiz



? **F281x Flash programming is done by:**
- ✓ **Executing algorithm code on the DSP.**

? **The algorithms must be configured for:**
- ✓ **The CPU frequency of the device.**

? **And must be executed from:**
- ✓ **Zero wait state SARAM**

? **The erase operation:**
- ✓ **Removes charge from the floating gates within a sector so all bits in the sector read back a 1.**

? **The program operation:**
- ✓ **Deposits charge on the floating gate to make specified single bits read back 0.**

? **The OTP cannot be:**
- ✓ **Erased**

Section 2 – Development Solutions

Learn about programming solutions for the firmware development and prototype phase.

How can I easily program the Flash during firmware development?

How can I program a few prototypes on machines without CCS installed?

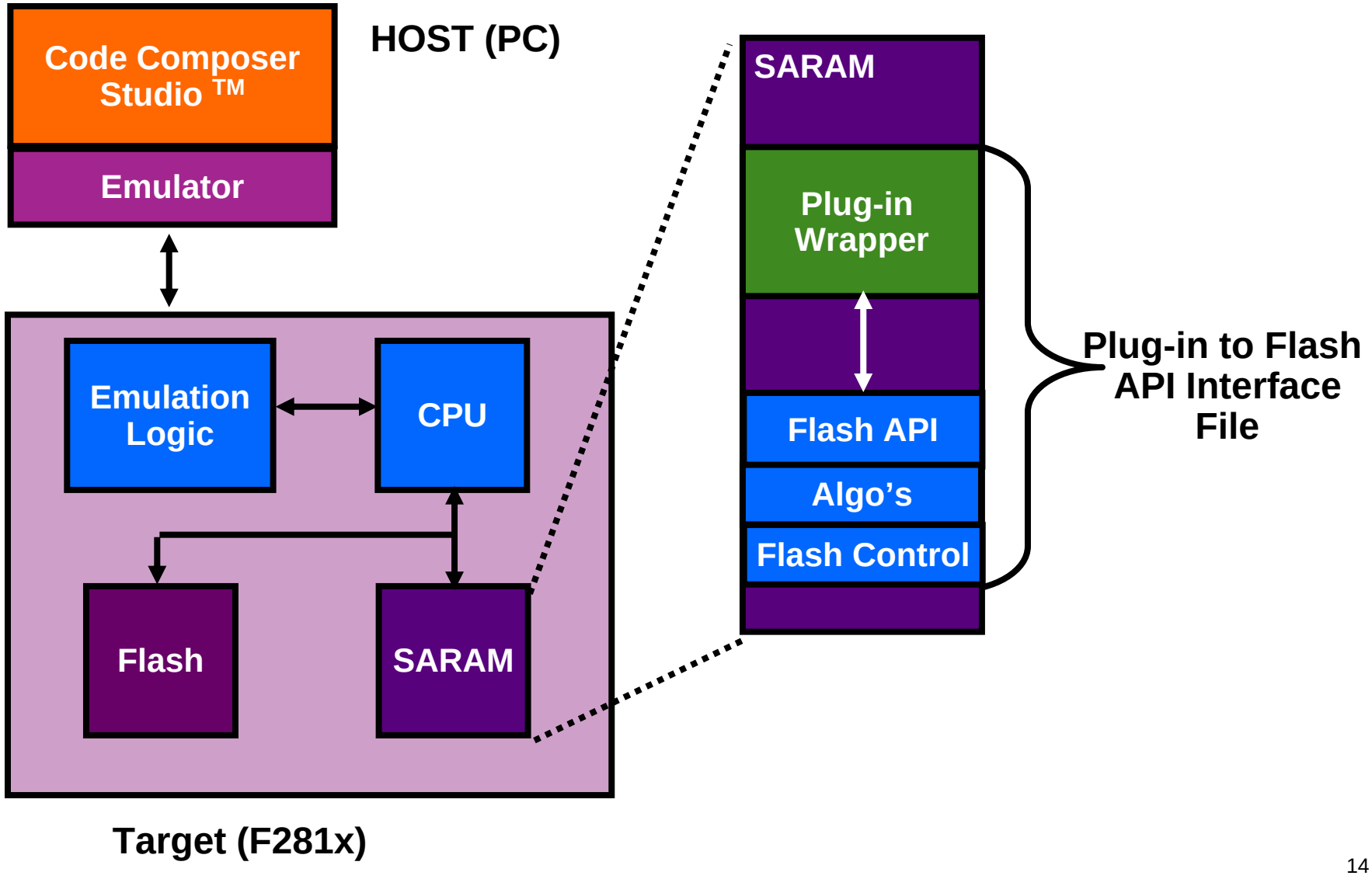
Development Solutions

- **TMS320C2000 Code Composer Studio™ on-chip Flash programmer plug-in**
- **SDFlash from Spectrum Digital Inc. (www.spectrumdigital.com)**

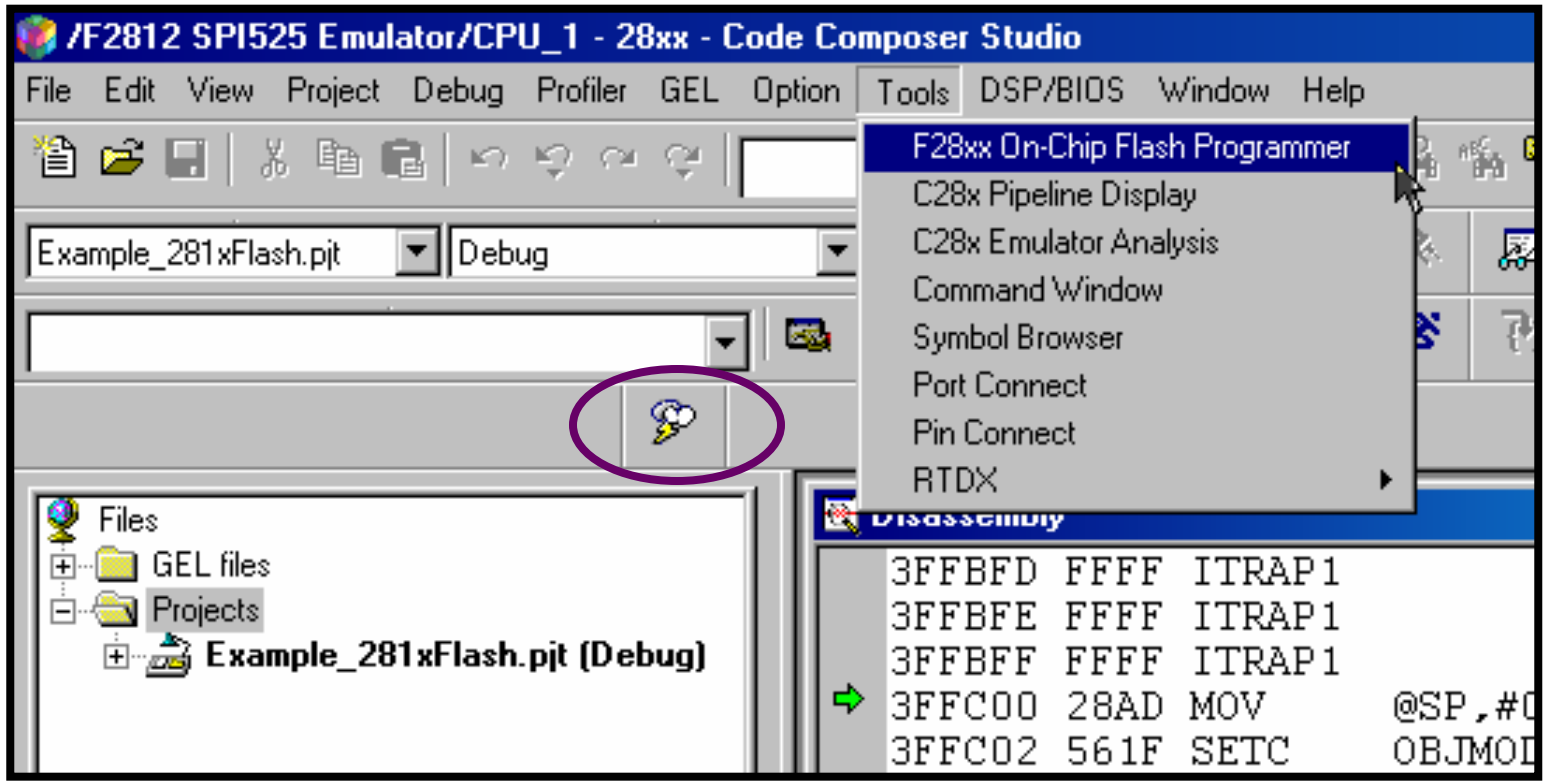
Code Composer Studio Plug-in

- Integrated Flash programming tool within the Code Composer Studio environment including on-line help.
- Developed specifically for the C2000 Flash devices and feature set.
- No need to close CCS and switch tools to program the device.
- Available for full CCS 2.2 and later via update advisor.

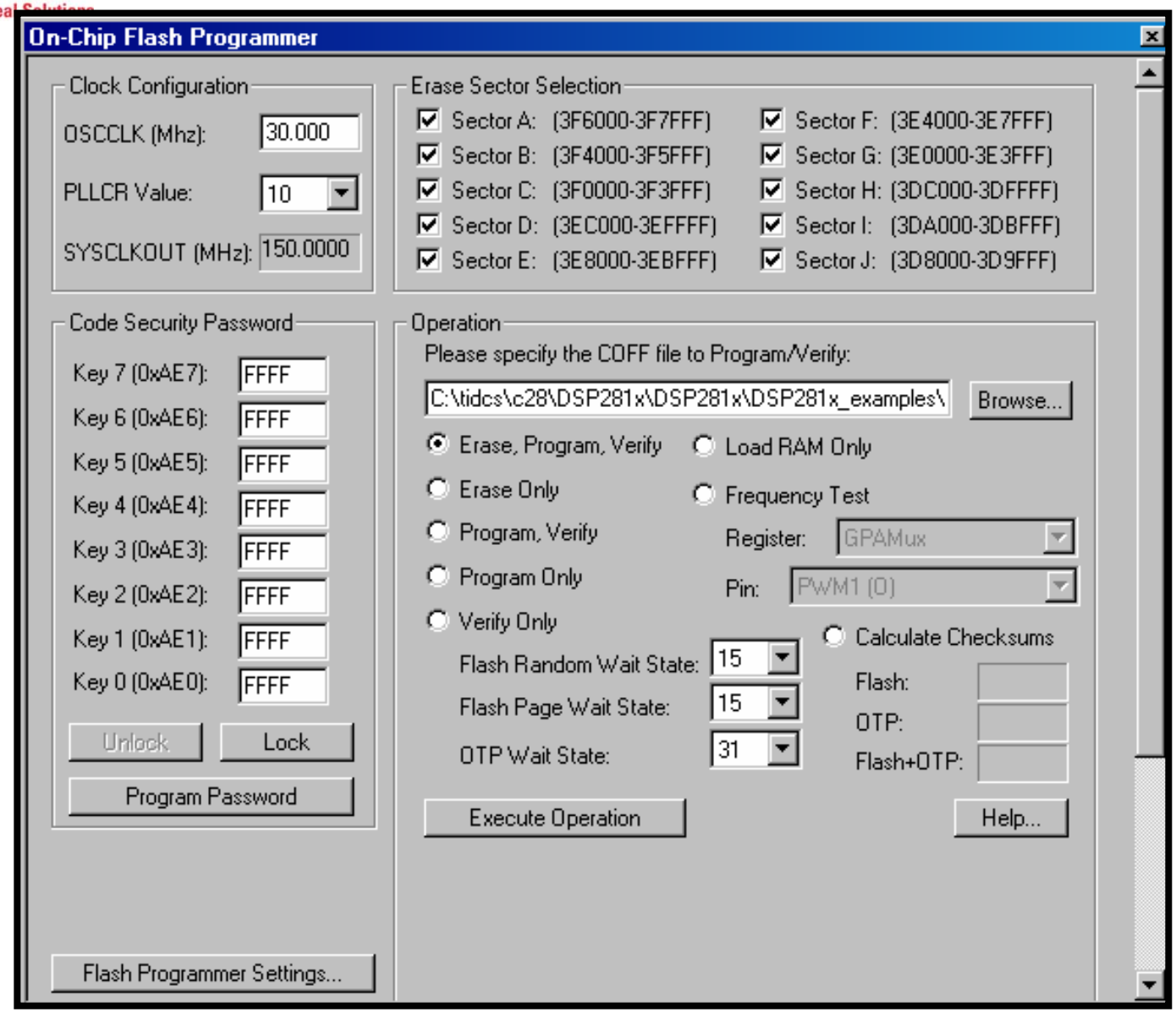
Code Composer Studio Plug-in



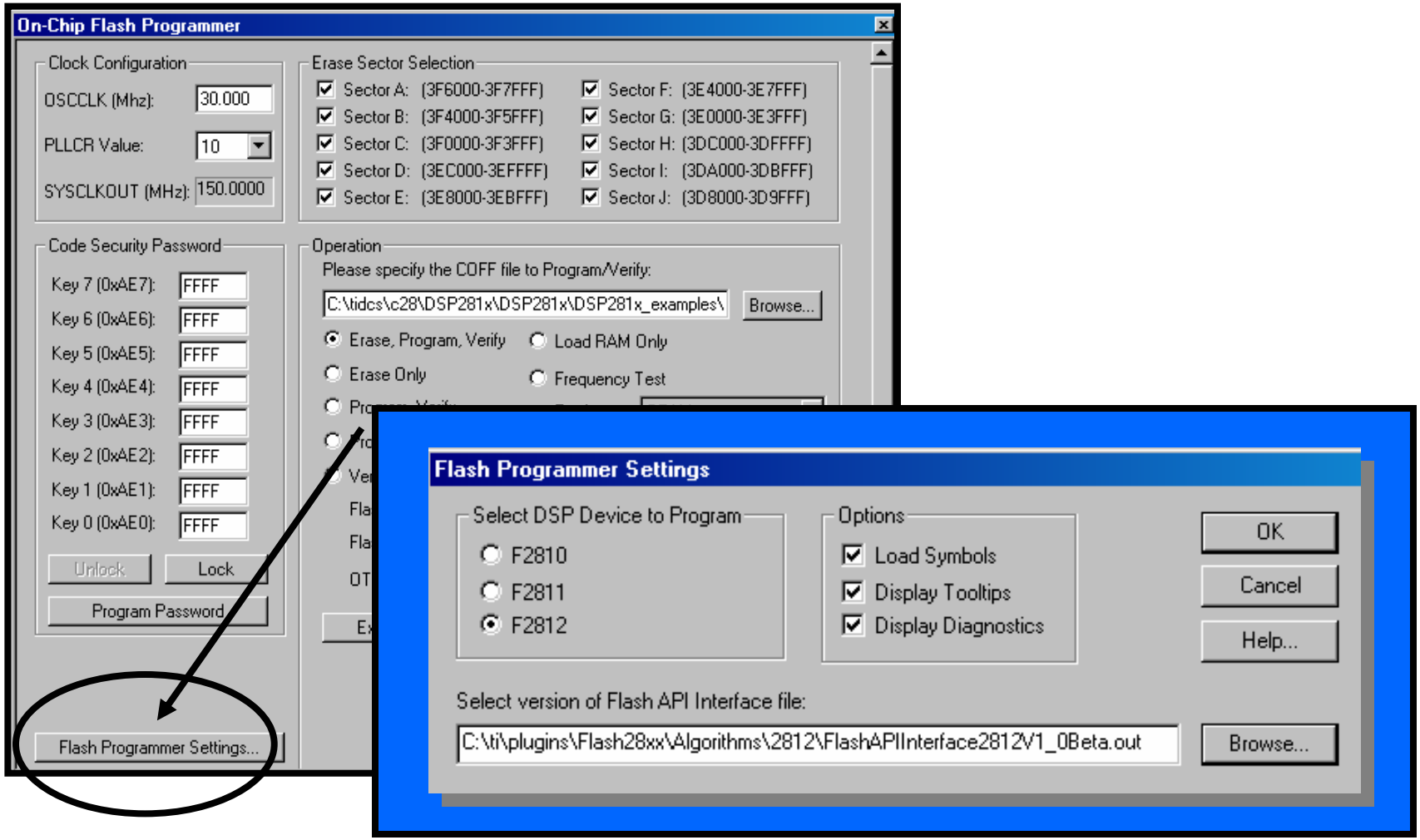
Code Composer Studio Plug-in



Code Composer Studio Plug-in

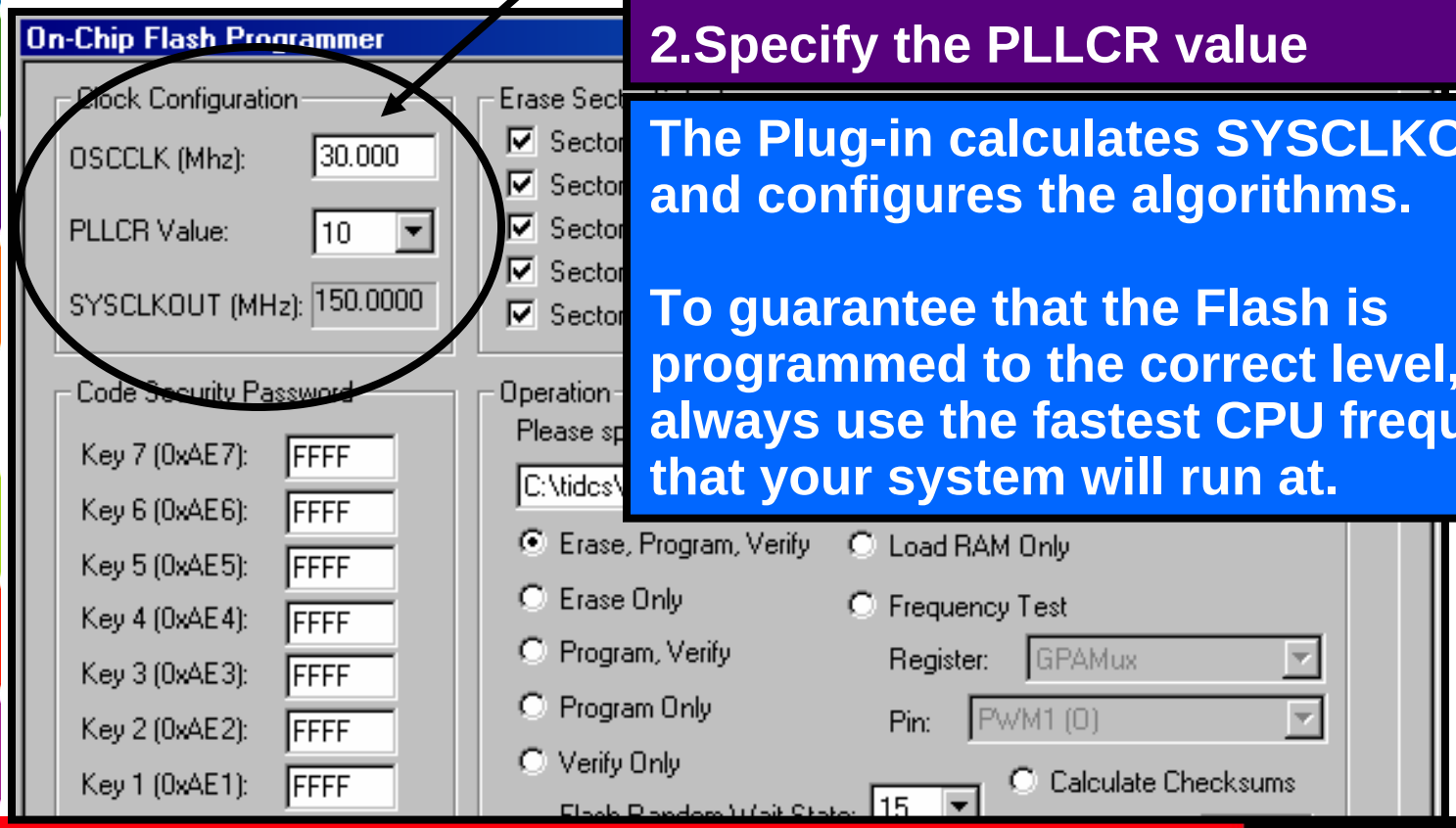


Code Composer Studio Plug-in



Code Composer Studio Plug-in

How do I configure the algorithm for my system's operating frequency?



1.Specify the input clock frequency

2.Specify the PLLCR value

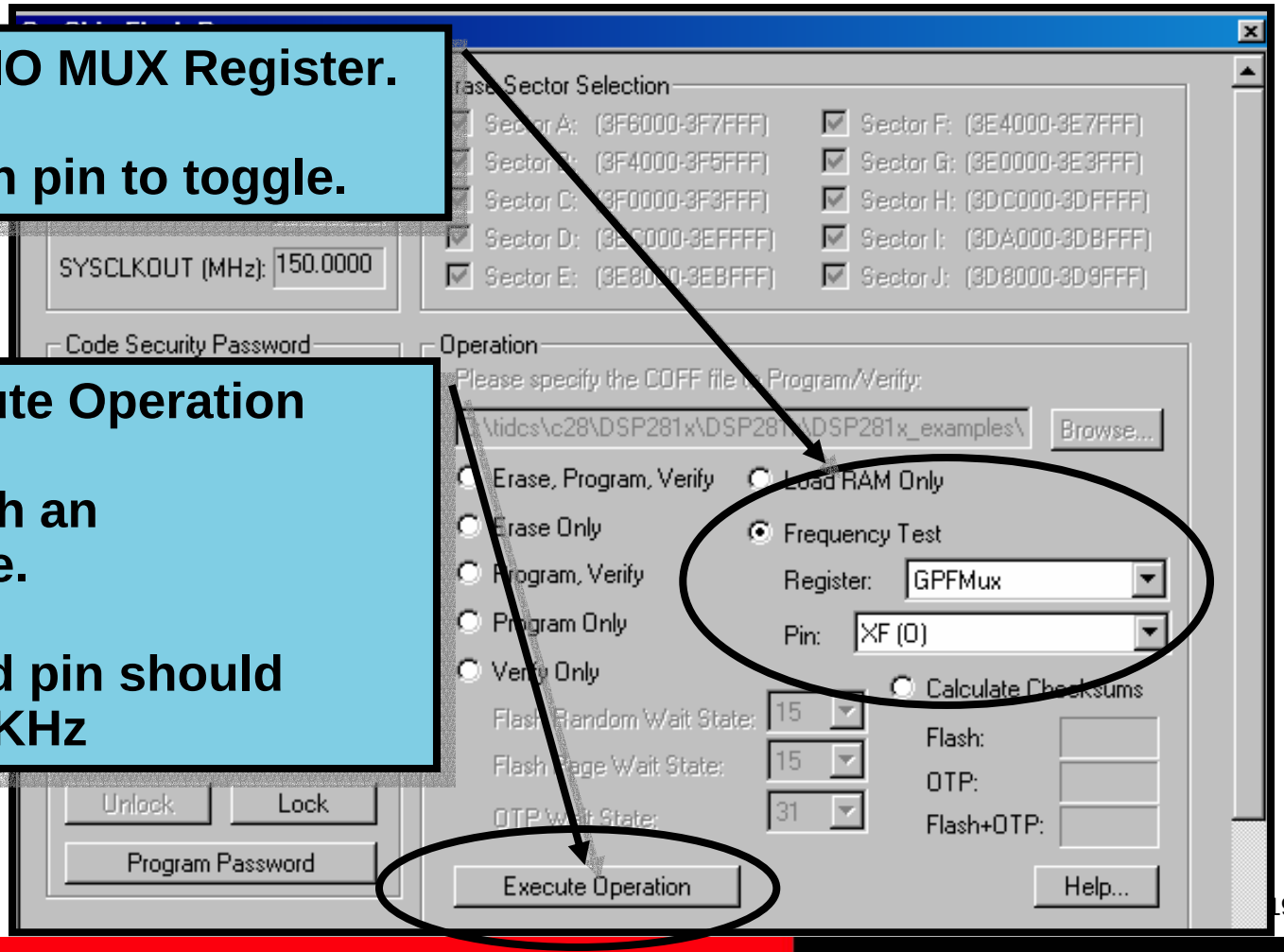
The Plug-in calculates SYSCLKOUT and configures the algorithms.

To guarantee that the Flash is programmed to the correct level, always use the fastest CPU frequency that your system will run at.

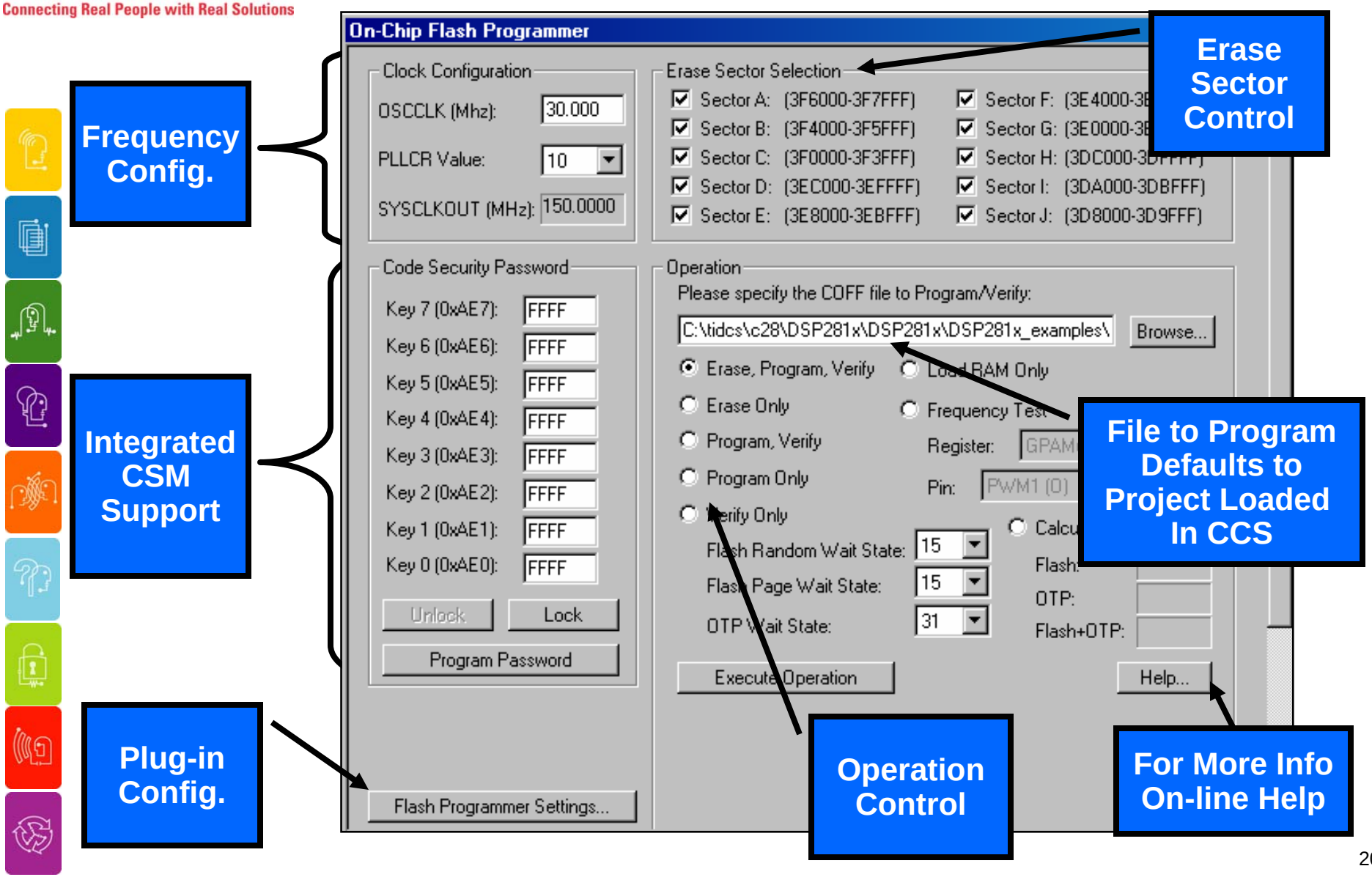
Code Composer Studio Plug-in

How can I test the frequency configuration?

1. Select a GPIO MUX Register.
2. Select which pin to toggle.
3. Press Execute Operation
4. Observe with an oscilloscope.
5. The selected pin should toggle at 10KHz



Code Composer Studio Plug-in

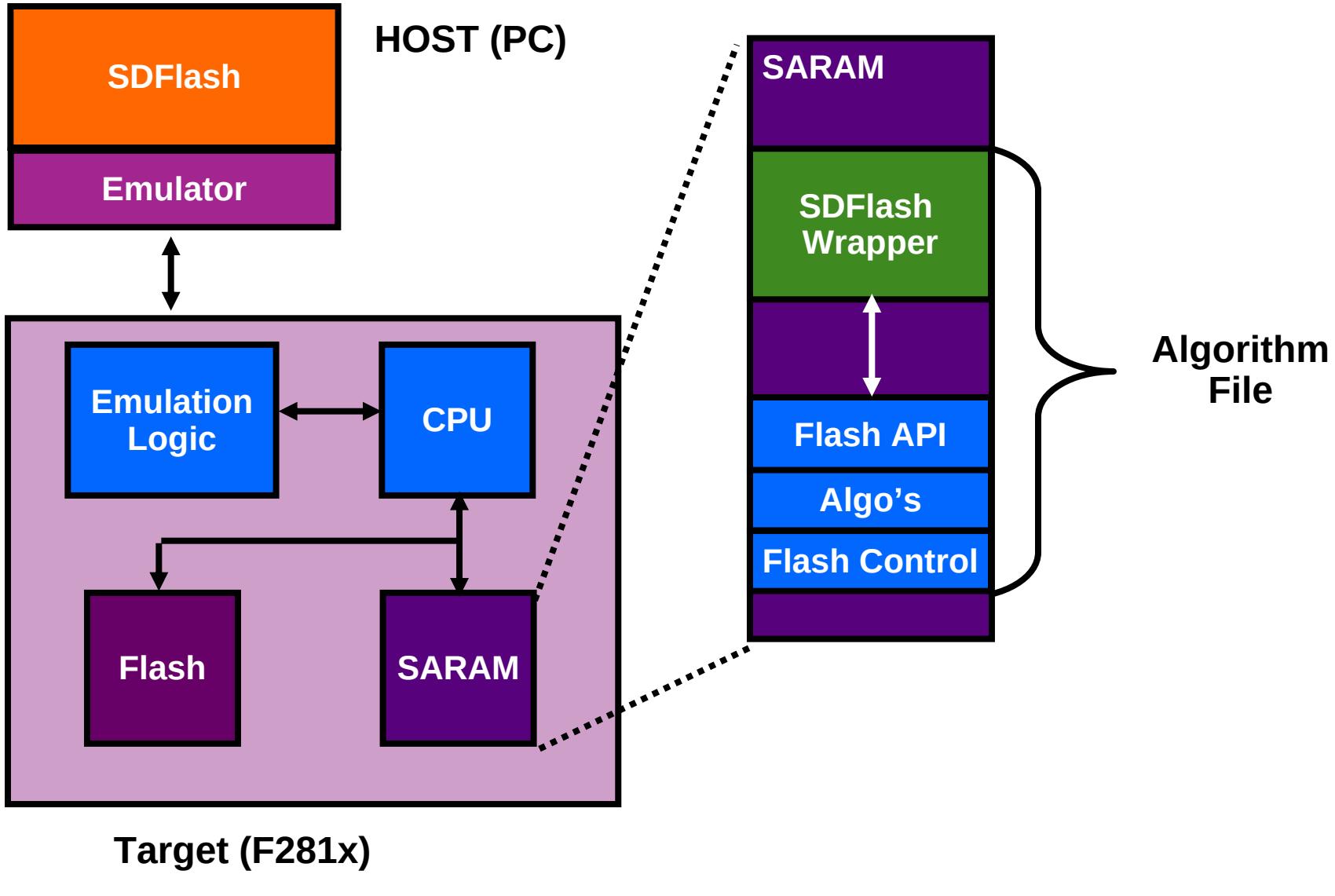


SDFlash

- SDFlash is a stand alone generic Flash programming interface from Spectrum Digital Inc. (http://www.spectrumdigital.com)
- SDFlash does not require Code Composer Studio. Only the Spectrum Digital JTAG emulation driver is required for JTAG programming.
- RS232 programming with example communication interface code is also available as of SDFlash V1.60.

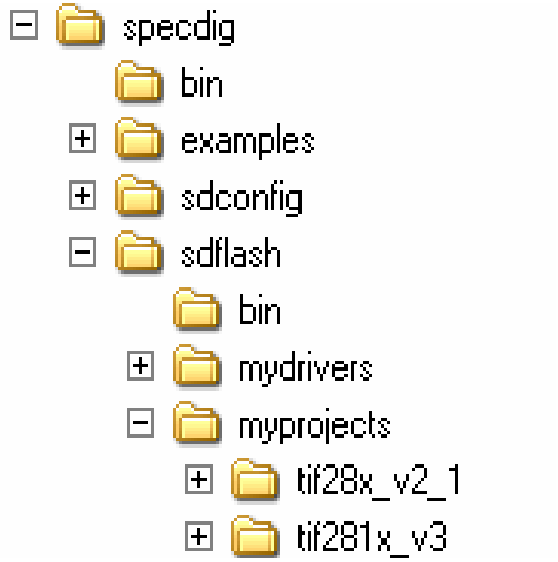
http://www.spectrumdigital.com/drivers/
/download.cgi?file=docstore/LatestC2000Tools.htm

SDFlash Stand Alone Programmer



Downloading SDFlash

How can I get SDFlash?

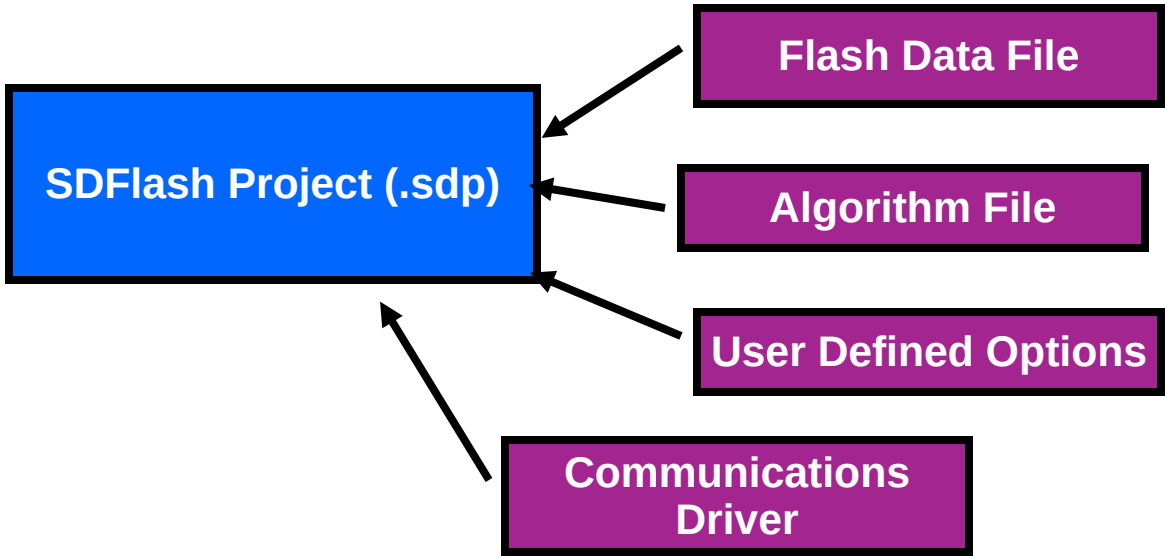


1. **SDFlash GUI interface:** Download the C2000 emulation drivers from Spectrum Digital.
2. **SDFlash algorithm files:** Download the TI supplied algorithm files from the Spectrum Digital website and unzip them into the sdflash\myprojects directory.

http://www.spectrumdigital.com/drivers/
/download.cgi?file=docstore/LatestC2000Tools.htm

The SDFlash Project

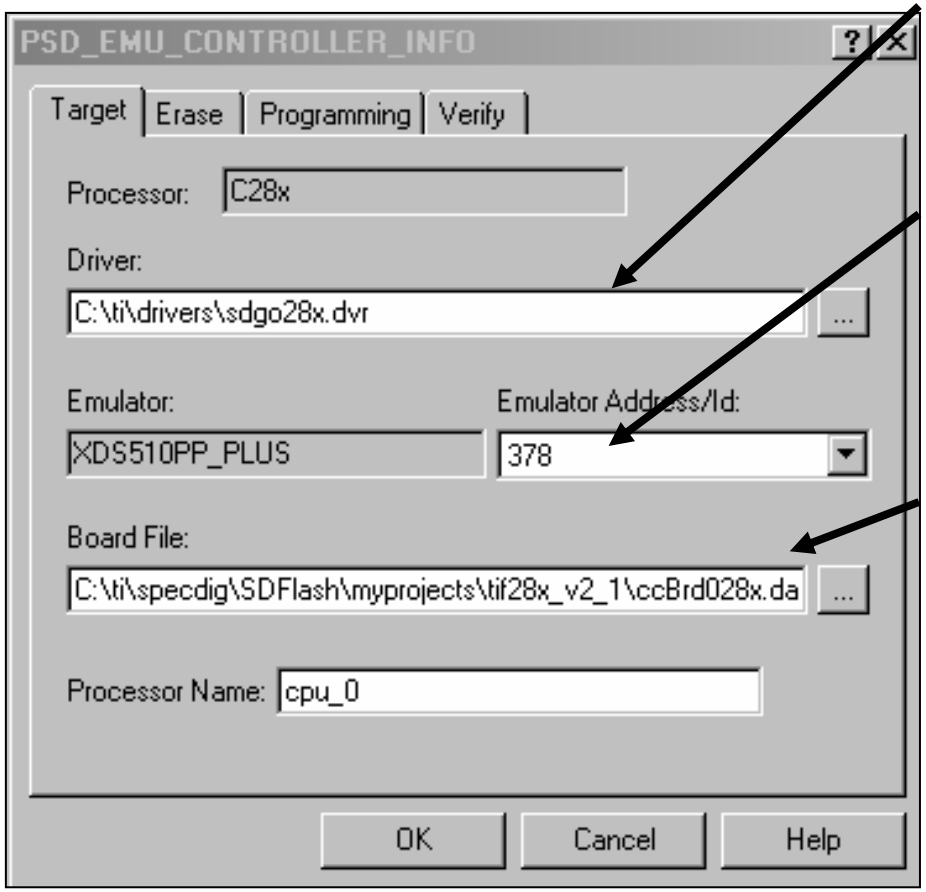
An SDFlash project is a text file that is used to store your Erase and Program settings.



You can view and edit SDFlash project contents through the SDFlash GUI interface.

Sample projects are included with the algorithm files.

SDFlash Setup - Target



SD Emulation Driver Downloaded with SDFlash

JTAG Port Address Setup with SDConfig

Board File Provides information on what Kind of devices are on the JTAG scan chain

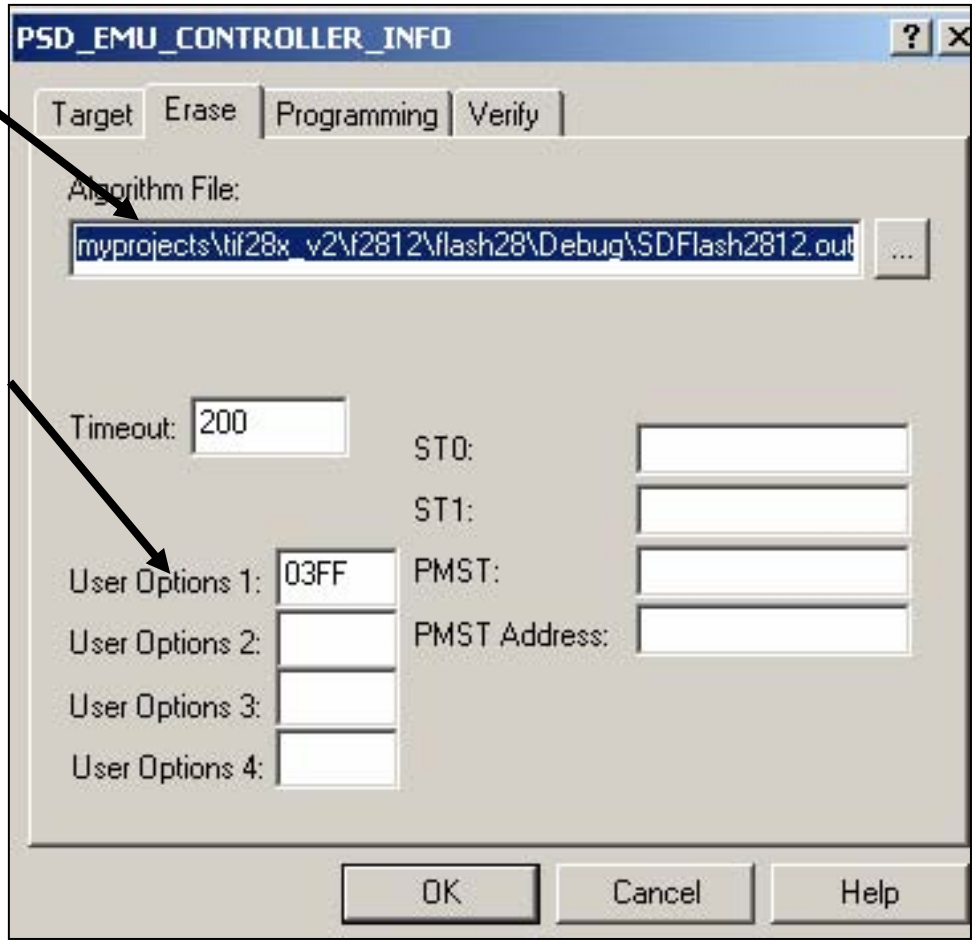
SDFlash Setup - Erase

Algorithm File
SDFlash Wrapper + Flash API

User Option 1

For 281x used to specify a sector mask for the erase operation.

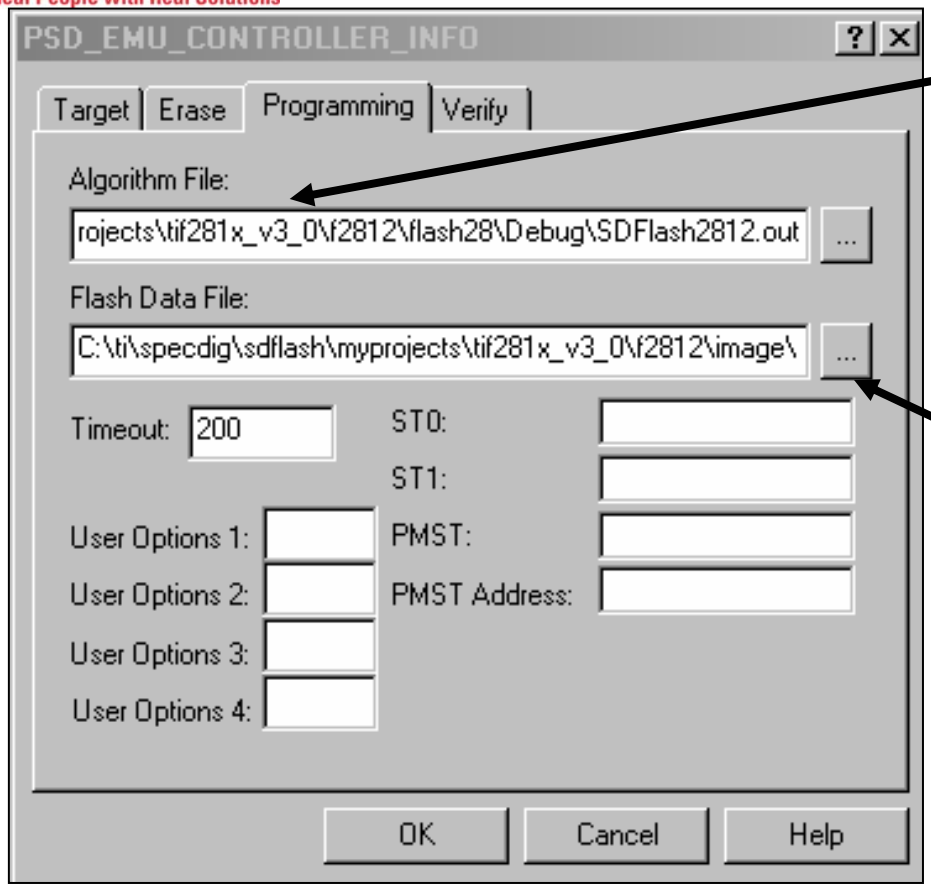
Functionality of user options is algorithm specific.



Erase Sector Mask

Bit 9			Bit 3	Bit 2	Bit 1	Bit 0
.....	Sec J	.....	Sec D	Sec C	Sec B	Sec A

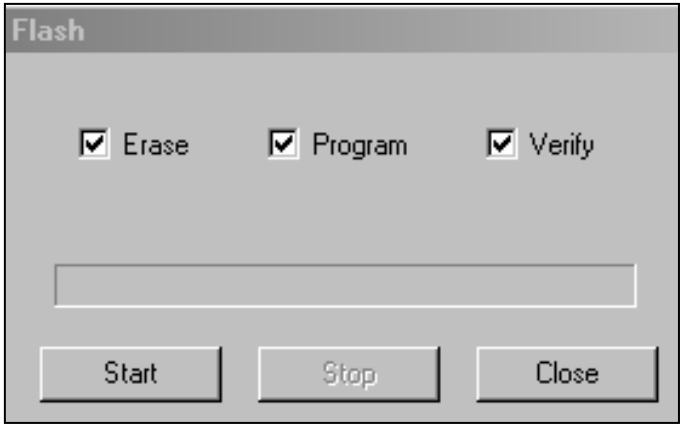
SDFlash Setup - Program



Algorithm File
SDFlash Wrapper + Flash API

Supplied by TI
Download from SD's website

Flash Data File
.out file to be programmed into the Flash/OTP

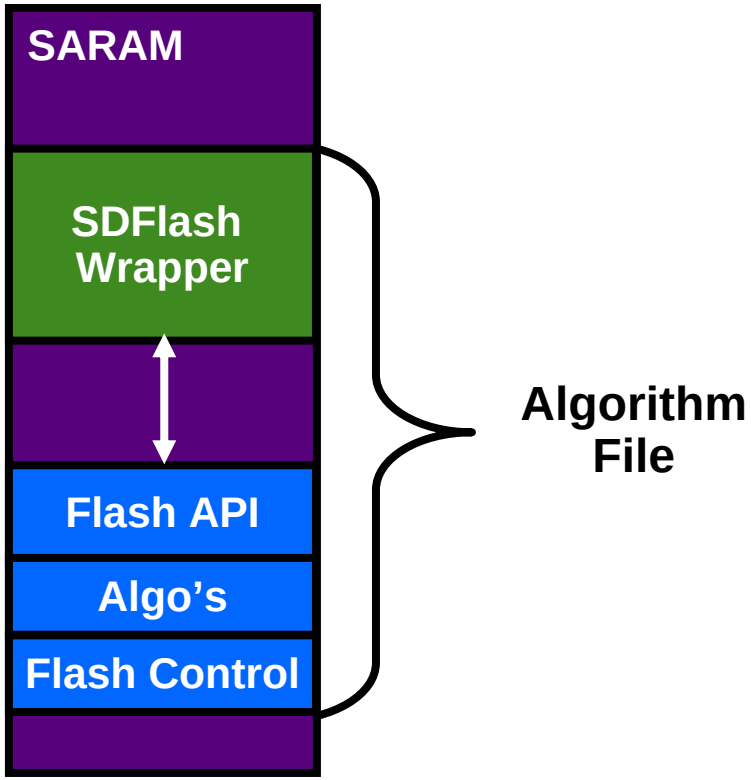


Easy to use interface allows you to perform desired operations

SDFlash Frequency Configuration

What about frequency configuration?

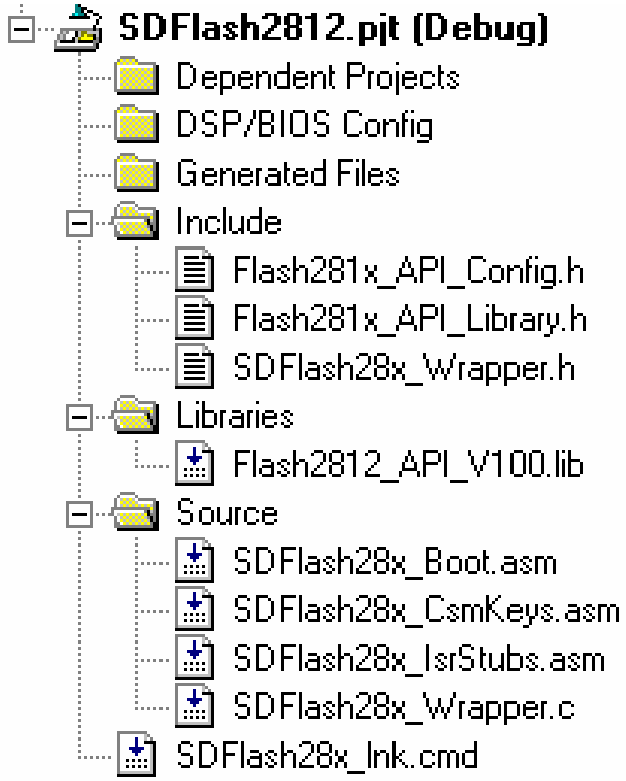
SDFlash JTAG Algorithm File CCS Project



SDFlash Frequency Configuration

What about frequency configuration?

SDFlash JTAG Algorithm File CCS Project



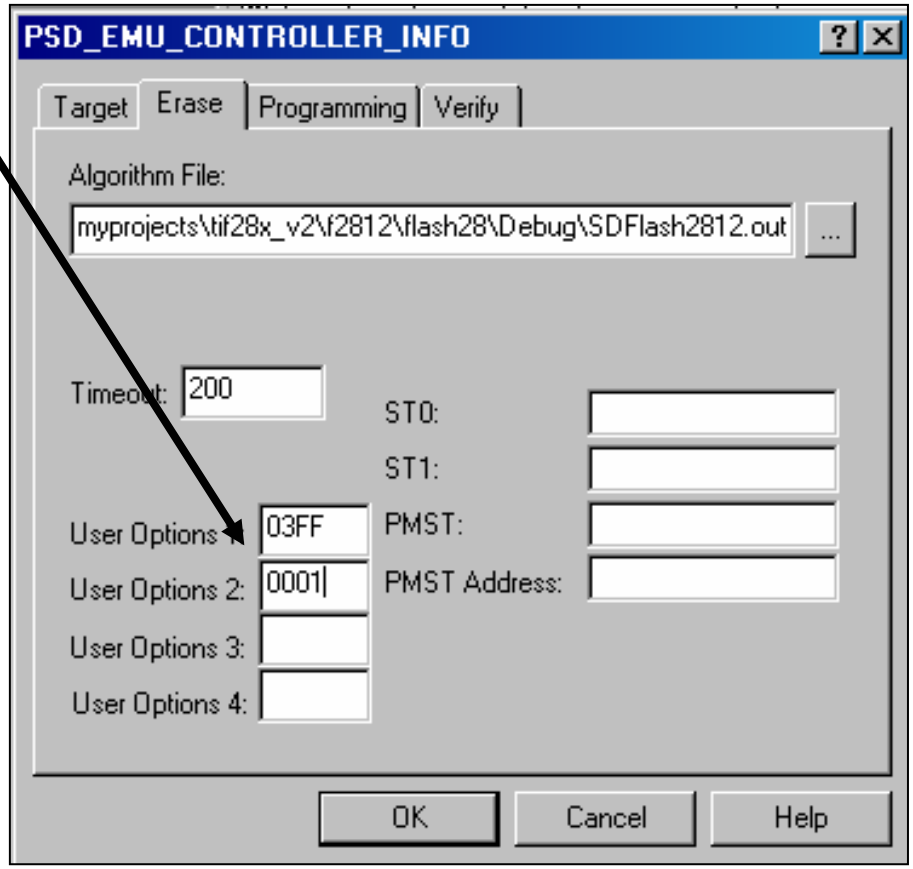
1. Specify the PLLCR setting in SDFlash28x_Wrapper.h
2. Specify the CPU frequency in Modify Flash281x_API_Config.h
3. Rebuild the algorithm file.
4. Close Code Composer Studio.
5. Run the frequency toggle test from to verify proper frequency configuration!

SDFlash Frequency Configuration

Erase User Option 2

Used to run the frequency pin toggle test

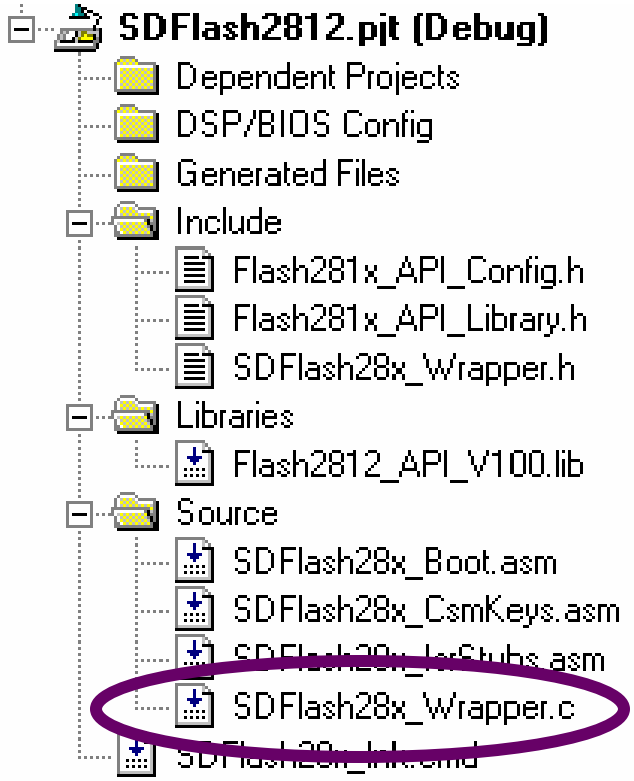
The value indicates which pin to toggle:



	Pin Toggled
blank	Test not run
0000	Test not run
0001	GPIOF14_XF
0002	GPIOA0_PWM1
0003	GPIOF4_SCITXDA
0004	GPIOG4_SCITXDB
0005	GPIOF12_MDXA
> 0006	Test not run

Customizing SDFlash

SDFlash JTAG Algorithm File CCS Project



You can modify the SDFlash wrapper to perform custom operations before or after calling the Flash API.

Example:

Before verify, perform a checksum on the Flash contents and compare it against a golden value.

Section 3 – Custom Solutions

Understand how to develop custom programming solutions.

How can I add Flash programming to my embedded system?

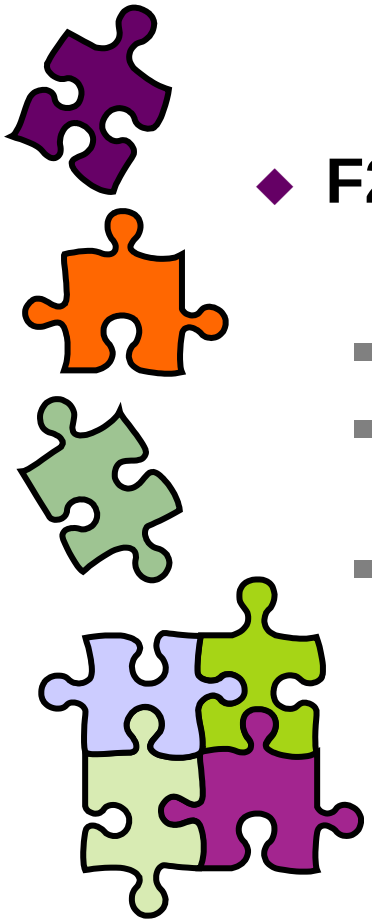
How do I create custom programming solutions?

How can I perform updates in the field?

What resources are there for production programming?

Custom Programming

How can I create my own programming solution?



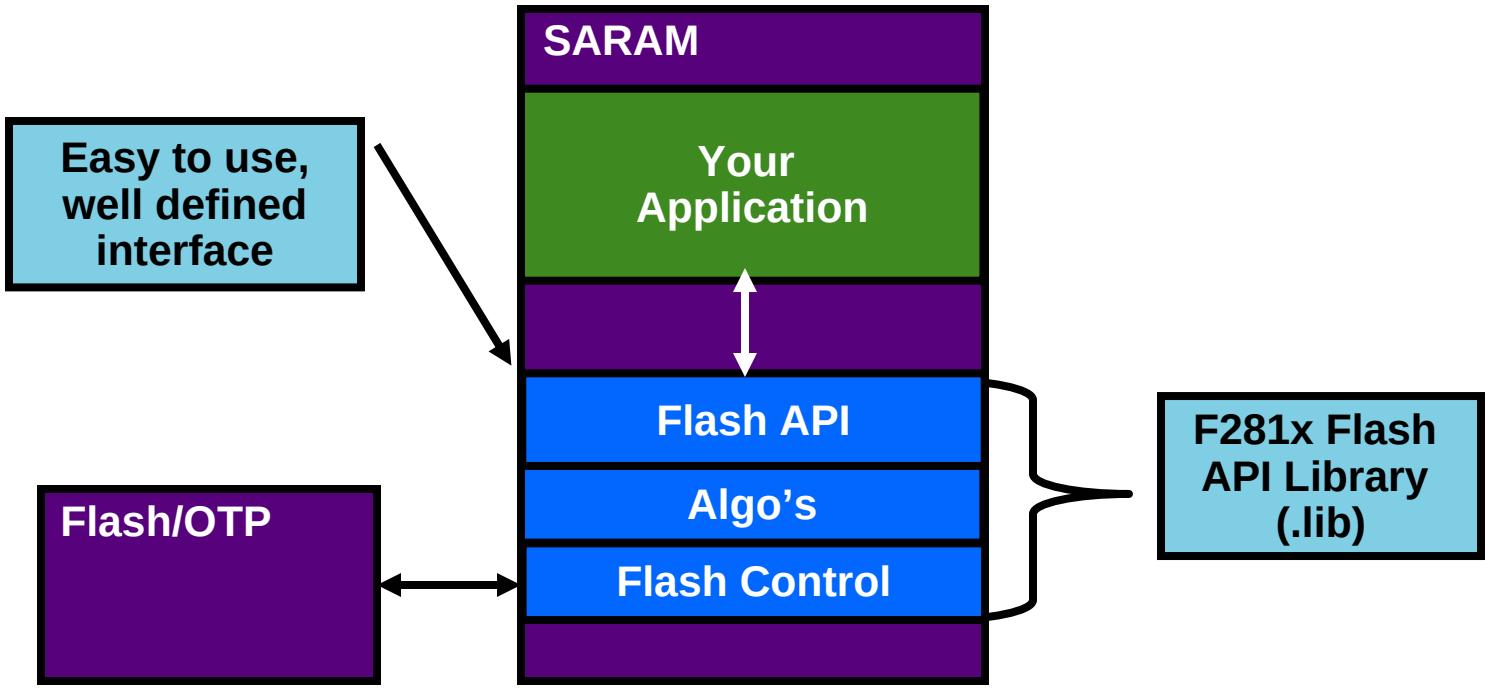
- **F281x Flash API (SPRC125)**
  - **Used by both the CCS plug-in and SDFlash.**
  - **Allows you to create custom programming solutions (example: RS232, eCAN)**
  - **You can also add Flash programming to your embedded application.**

http://focus.ti.com/docs/toolsw/folders/print/c28xflashtools.html

F281x Flash API

What is the Flash API Library?

- The Flash API library consists of TI supplied Flash programming algorithms with a well defined and easy to use interface.



Flash API Integration

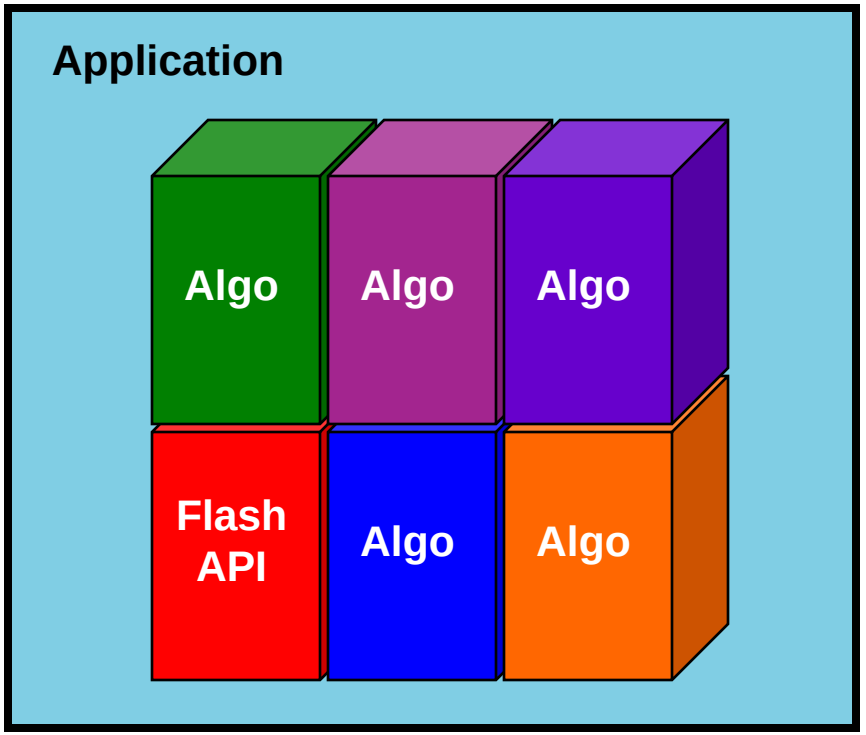
F281x Flash API

Ready to go Algorithms

+

Easy Integration

Shorter Development Cycle



F281x Flash API Function Calls

Erase specified sectors:

Uint16 Flash2812_Erase(SectorMask, &FStatus)

- **SectorMask: Which sectors to erase.**
- **&Fstatus: Pointer to status structure.**

Program code and data into Flash/OTP:

Uint16 Flash2812_Program(&FlashAddr, &BuffAddr, Length, &FStatus)

- **&FlashAddr: Pointer to first Flash/OTP address to program.**
- **&BuffAddr: Pointer to the buffer of data/code to program.**
- **Length: Number of 16-bit words to be programmed.**
- **&Fstatus: Pointer to the Flash status structure.**

F281x Flash API Function Calls

Verify proper algorithm frequency configuration:

Uint16 Flash2812_ToggleTest (&MuxReg, &ToggleReg, Mask)

- **&MuxReg: Pointer to a GP I/O MUX register.**
- **&ToggleReg: Pointer to a GP I/O TOGGLE register.**
- **Mask: Mask indicating which pin to toggle.**

Verify values in Flash/OTP:

Uint16 Flash2812_Verify(&FlashAddr, &BuffAddr, Length, &FStatus)

- **&FlashAddr: Pointer to first location within the Flash/OTP**
- **&BuffAddr: Pointer to the buffer to compare against.**
- **Length Number of 16-bit words to compare.**
- **&FStatus: Pointer to the Flash status structure.**

API Status Structure: FLASH_ST

What is the Flash status structure?

```
Uint16 Flash2812_Program(&FlashAddr, &BuffAddr, Length, &FStatus);
```

In Flash281x_API_Library.h

```
typedef struct {
   Uint32  FirstFailAddr;
   Uint16  ExpectedData;
   Uint16  ActualData;

}FLASH_ST;
```

In your application:

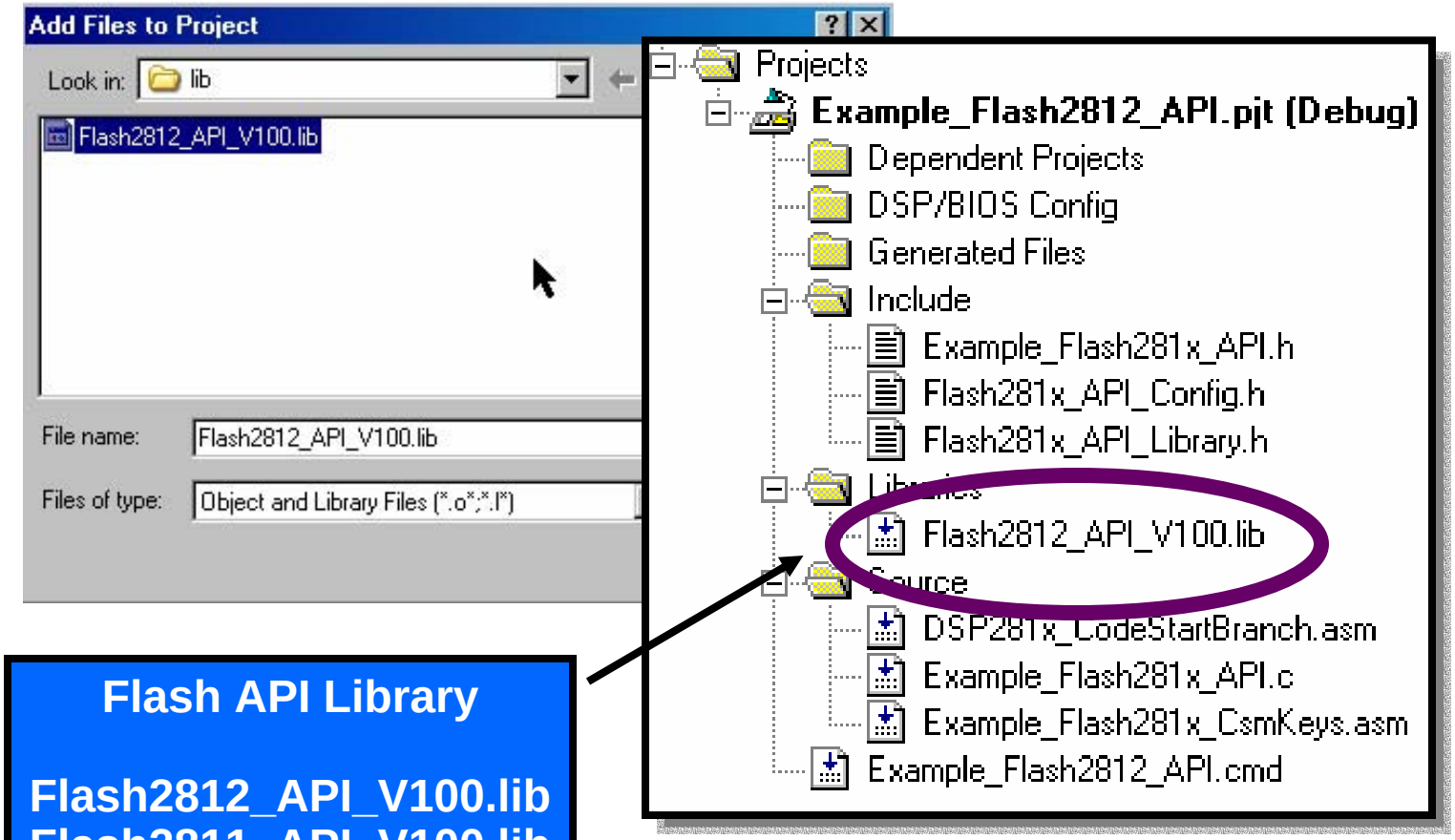
```
FLASH_ST FStatus
```

Working With the Flash API

- **To add embedded Flash programming to your project, you must make the following changes:**

1. **Add the Flash API Library to your project.**
2. **Include the Flash API header file in your source code.**
3. **Initialize the PLLCR and configure the Flash algorithms for the proper CPU frequency.**
4. **Execute the Flash API source is in single cycle SARAM.**
5. **Don't forget the Code Security Module**

Step 1: Add the Flash API Library



Flash API Library

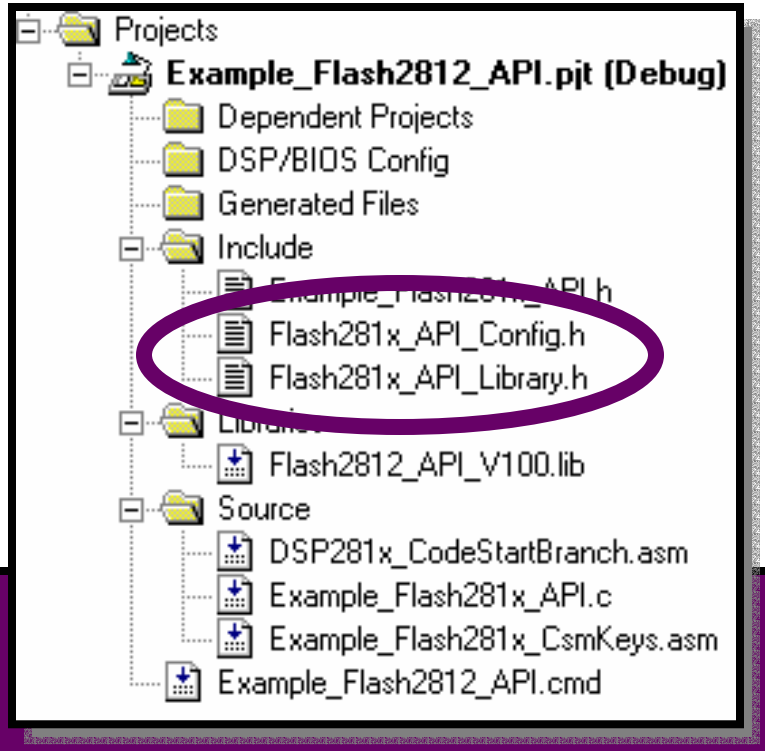
Flash2812_API_V100.lib
Flash2811_API_V100.lib
Flash2810_API_V100.lib

Step 2: Include the API Header File

Include the Flash API Header File in your application source code.

This file includes:

- **Function prototypes.**
- **Status structure definition.**
- **API error codes.**
- **API Config.h file.**



Your application:

```
/*---- Flash API include file -------------------*/
#include "Flash281x_API_Library.h"
```

Step 3: Configure the API For Your Operating Frequency

You must configure the API for the CPU operating frequency of your system.

Modify Flash281x_Config.h to specify the proper CPU frequency.

Flash281x_Config.h:

```
  #define CPU_RATE   6.667L      // for a 150MHz CPU
//#define CPU_RATE   7.143L      // for a 140MHz CPU
```

The CPU_RATE is used to calculate a scale factor that you will use to configure the algorithms for the correct CPU frequency

Flash281x_Config.h:

```
// Do not modify this line!!
#define SCALE_FACTOR 1048576.0L*((200L/CPU_RATE))
```

Step 3: Configure the API For Your Operating Frequency

Add code to your application to initialize the global variable Flash_CPUScale_Factor.

This variable is used by the API to scale critical timing loops.

```
Your application:

  Flash_CPUScale_Factor = SCALE_FACTOR;
```

Initialize the PLLCR register and wait until the PLL has stabilized.

```
Your application:

   *PLLCR = PLLCR_VALUE;
 // Wait 131072 cycles for PLL to lock
```

Step 4: Copy the API to SARAM

If the Flash API source is stored in Flash/OTP, then you must copy it to SARAM before making any calls to the API.

Assign symbols to the load start, load end and run start addresses of the API source in the linker .cmd file:

Your linker .cmd file:

```
Flash28_API:
{
    Flash2812_API_V100.lib(.econst)
    Flash2812_API_V100.lib(.text)
}       LOAD = FLASHD,
        RUN = RAML0,
        LOAD_START(_Flash28_API_LoadStart),
        LOAD_END(_Flash28_API_LoadEnd),
        RUN_START(_Flash28_API_RunStart),
        PAGE = 0
```

Step 4: Copy the API to SARAM

Use these symbols to copy the source from its load address in Flash to its run-time address in SARAM:

Flash281x_API_Library.h:

```
extern Uint16 Flash28_API_LoadStart;
extern Uint16 Flash28_API_LoadEnd;
extern Uint16 Flash28_API_RunStart;
```

Your application source:

```
// Copy the Flash API functions to SARAM
Example_MemCopy(&Flash28_API_LoadStart,
                &Flash28_API_LoadEnd,
                &Flash28_API_RunStart);
```

Step 5: Don't Forget the CSM!

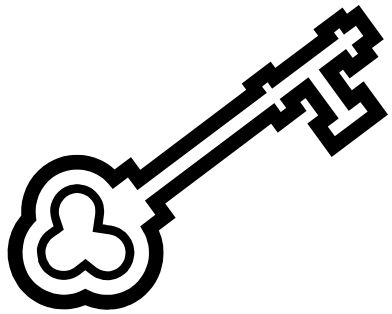
The Flash and OTP are protected by the Code Security Module (CSM).

In order to erase Flash or program the Flash/OTP:

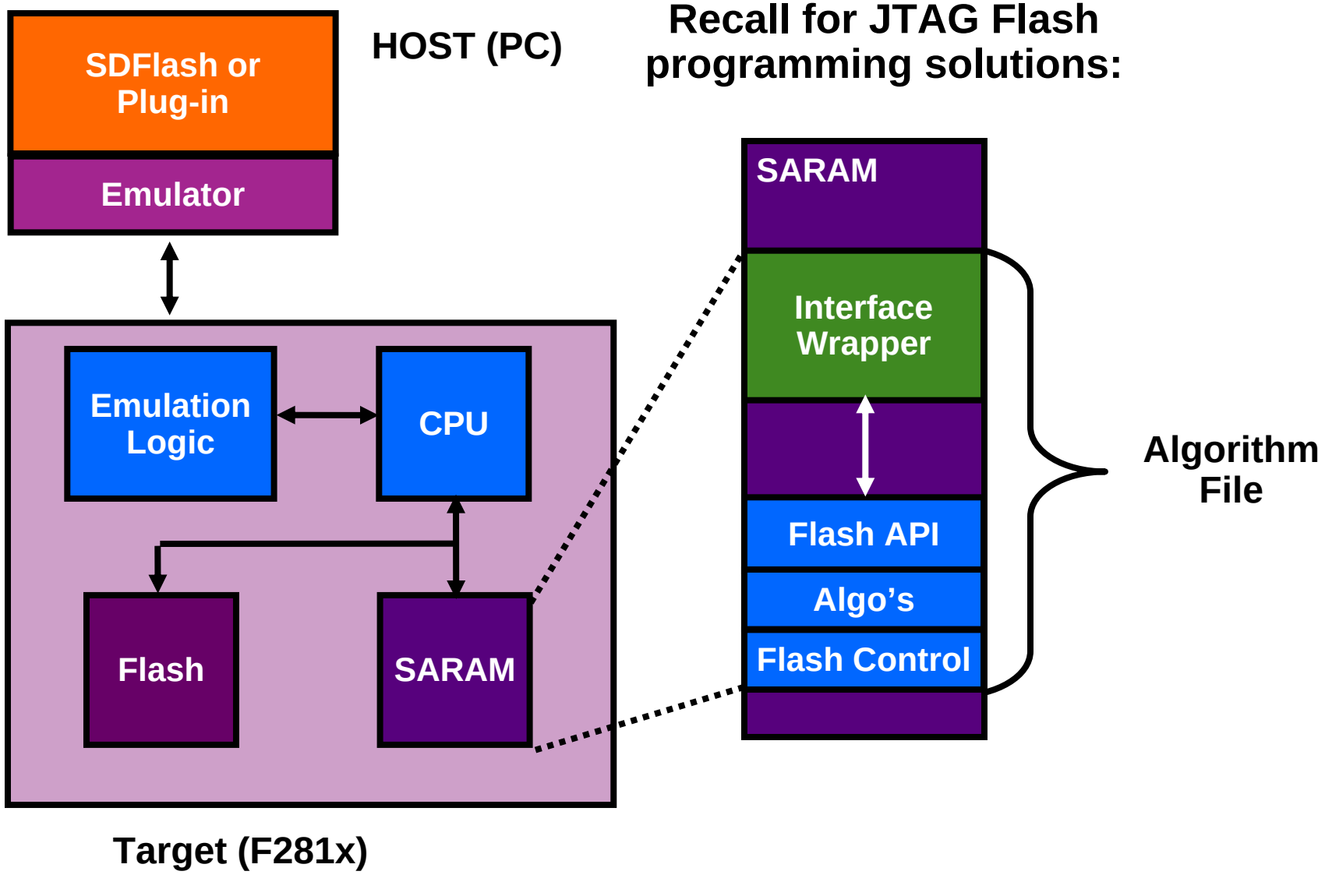
The CSM must be unlocked,

- OR -

The Flash API must be executed from secure SARAM memory.

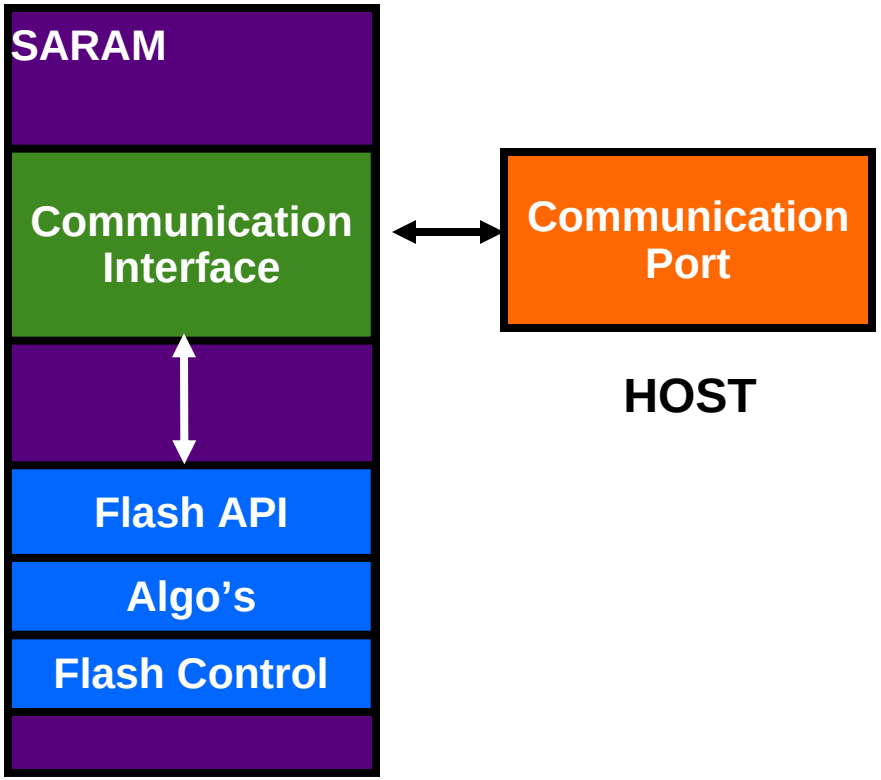


Custom Programming Solutions



Create Your Own Custom Programming Solutions

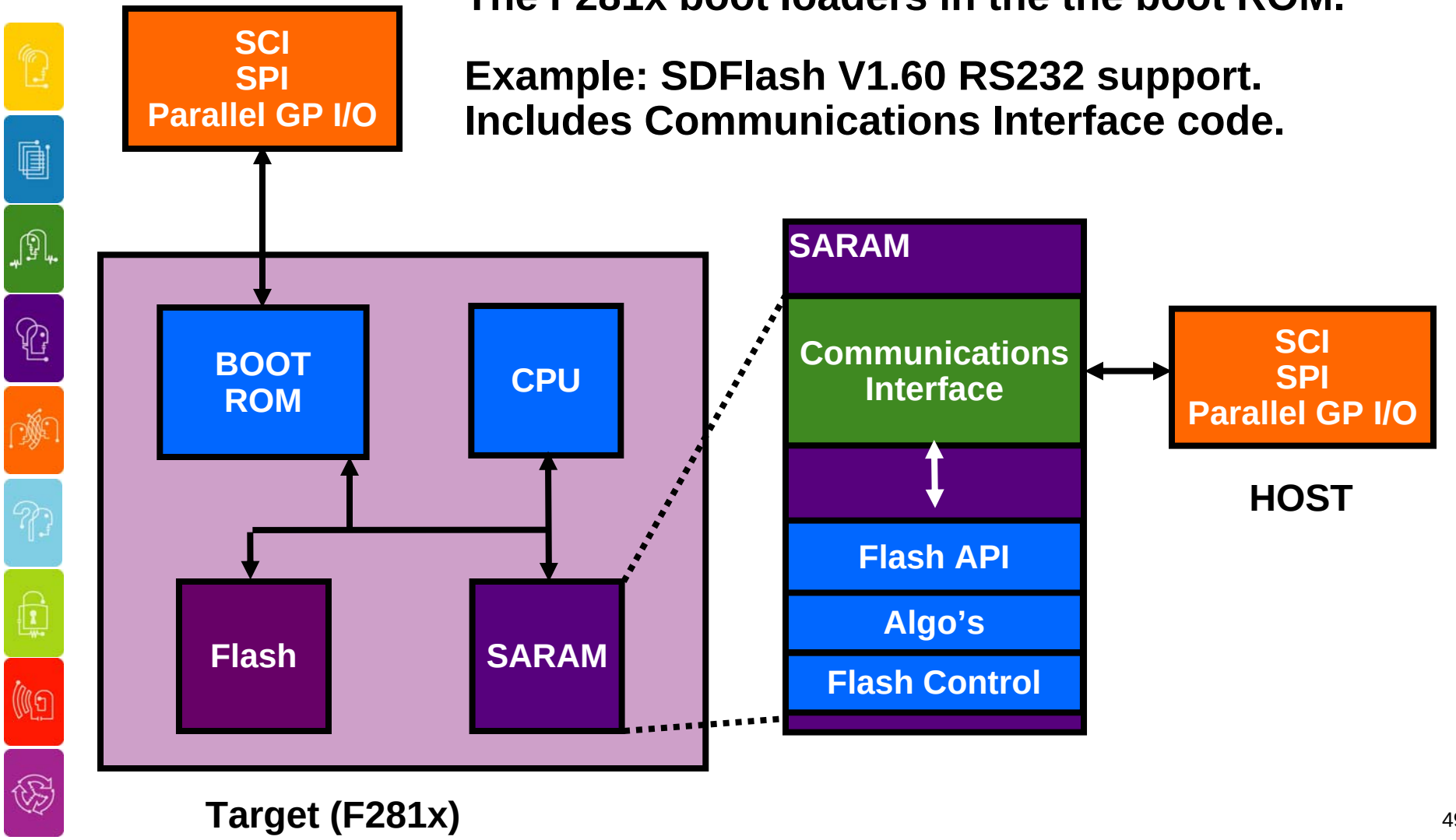
- You can extend this concept to other communication ports to create your own custom programming solutions.



Custom Programming Solutions

You can create custom programmers that use The F281x boot loaders in the the boot ROM.

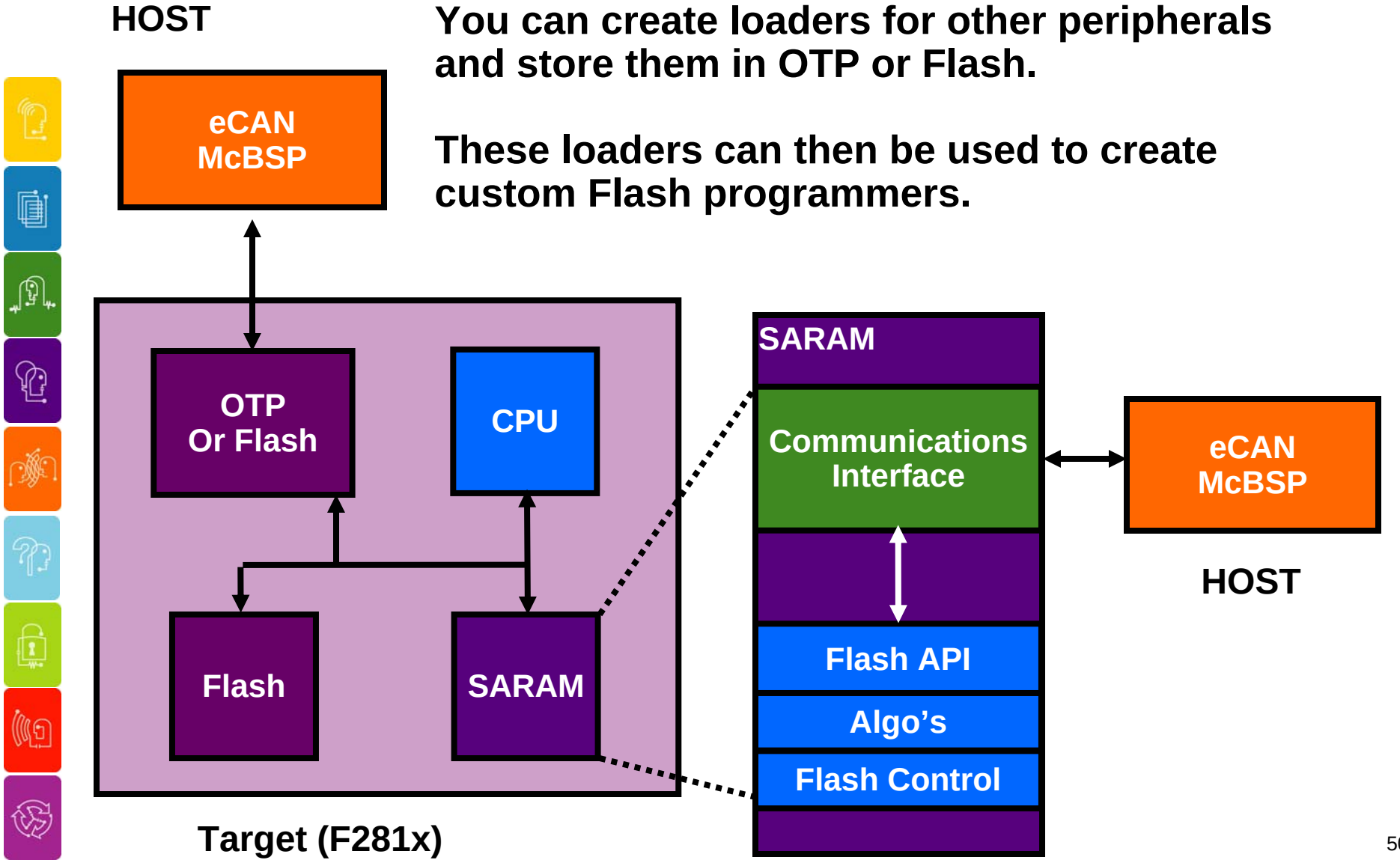
Example: SDFlash V1.60 RS232 support. Includes Communications Interface code.



Custom Programming Solutions

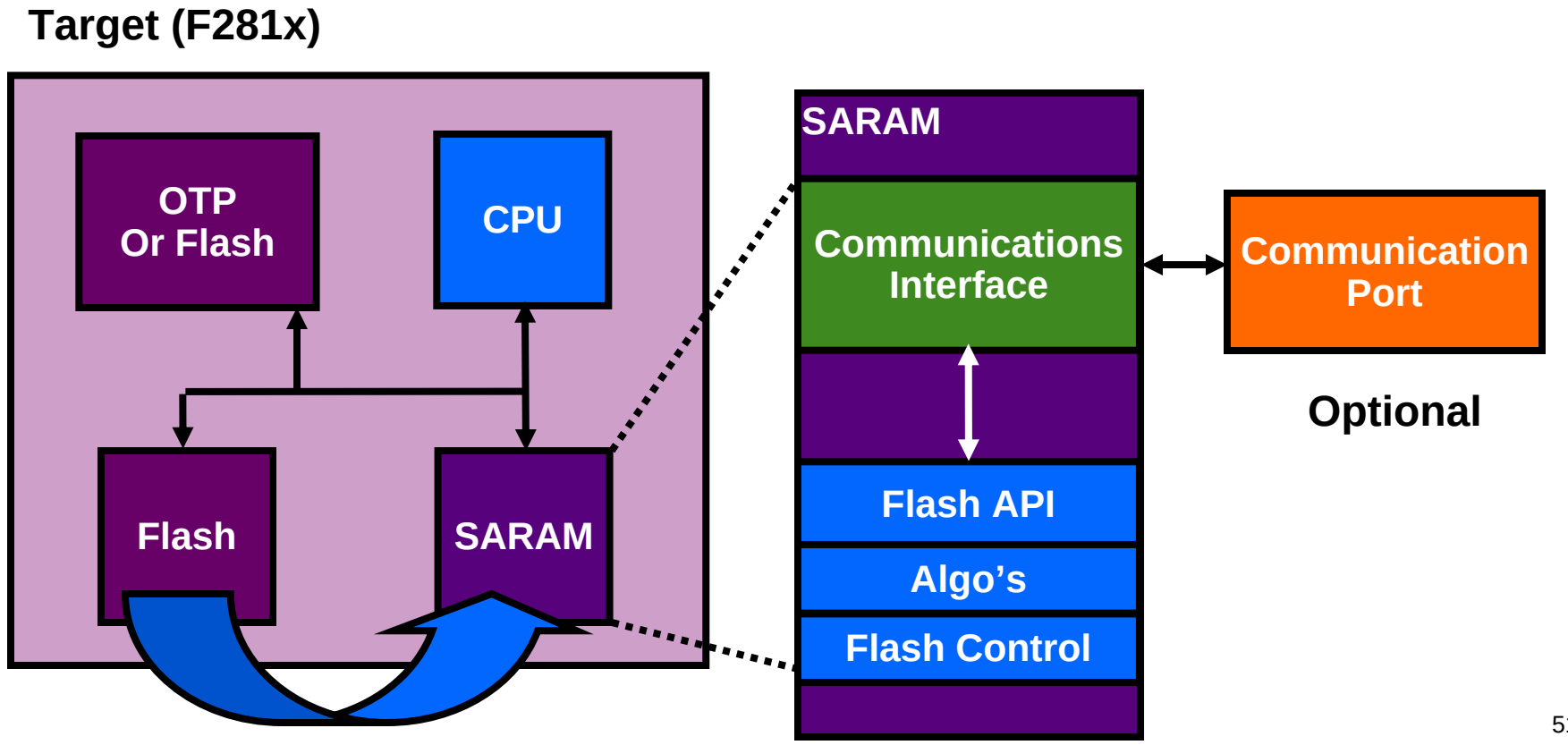
You can create loaders for other peripherals and store them in OTP or Flash.

These loaders can then be used to create custom Flash programmers.



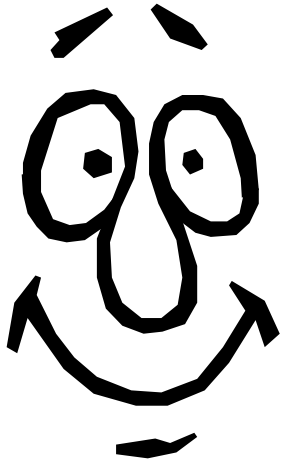
Custom Programming Solutions

The API can also be stored directly in Flash or OTP with your firmware and later copied to SARAM to perform Flash updates.



Embedded Flash Programming

A few Flash Programming Do's:



- ✓ Execute the algorithms from single cycle memory.
- ✓ Do execute the algorithms at the highest CPU frequency that your CPU will run at in the system.
- ✓ Configure the API for the correct CPU frequency.
- ✓ Do verify the frequency configuration.
- ✓ Unlock the CSM or execute the algorithms from secured memory.

Embedded Flash Programming

A Few Flash Programming Don'ts:

- ✓ Break any of the Do's.
- ✓ Do not run the algorithms from wait stated memory.
- ✓ Do not interrupt the algo's before completion.
- ✓ Do not expect to execute code or read from Flash/OTP while programming or erasing.

Resources: Large Scale Programming

TI Partners supporting C2000 Flash programming solutions include:

- **Data I/O www.dataio.com**
- **BP Microsystems www.bpmicrosystems.com**
- **Local Distributor**

Summary

- Flash programming can occur in every phase of the development cycle.
- C2000 Code Composer Flash Plug-in
  - Integrated way to quickly program the Flash during the development cycle.
  - Developed specifically for the F281x family of DSPs.
- SDFlash
  - Generic stand-alone interface from Spectrum Digital
  - Uses TI supplied algorithms to program the F281x DSPs.
  - SDFlash is available for both JTAG and RS232 programming.
- Flash API library.
  - Used by both the Plug-in and SDFlash.
  - Can be used to create custom programming solutions.

Control Track

TMS320F281X Flash Programming Solutions

Texas Instruments