# Module --

# High-Level Description

This module outlines the functionality of the system startup functions of the TMS570. This code is intended to be run starting at the reset vector.

# Figures

## Diagram – Function Data Sharing

This diagram shows all data that is shared between functions within the module.

No Shared Data

# Variable Data Dictionary

For details on module input / output variable, refer to the Data Dictionary for the application. Input / output variable names are listed here for reference.

|  |  |  |
| --- | --- | --- |
| Module Inputs | Module Outputs | |
| ResetCause\_Cnt\_Enum | | ResetCause\_Cnt\_Enum |
|  | |  |

## Module Internal Variables

This section identifies the name, range and resolutions for module specific data created by this module. If there are no range restrictions on the variable, the term “FULL” is placed into the table for legal range.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Variable Name | Resolution | Legal Range  (min) | Legal Range  (max) | Software Segment |
| <None> |  |  |  |  |
|  |  |  |  |  |

### User defined typedef definition/declaration

This section documents any user types uniquely used for the module.

|  |  |  |
| --- | --- | --- |
| Typedef Name | Element Name | Value |
| enum systemClockSource | SYS\_OSC | 0 |
|  | SYS\_PLL1 | 1 |
|  | SYS\_EXTERNAL1 | 3 |
|  | SYS\_LPO\_LOW | 4 |
|  | SYS\_LPO\_HIGH | 5 |
|  | SYS\_PLL2 | 6 |
|  | SYS\_EXTERNAL2 | 7 |
|  | SYS\_VCLK | 9 |

# Constant Data Dictionary

## Calibration Constants

This section lists the calibrations used by the module. For details on calibration constants, refer to the Data Dictionary for the application.

|  |
| --- |
| Constant Name |
| <None> |
|  |

## Program(fixed) Constants

### Embedded Constants

All embedded constants whose values are provided in Eng units will be evaluated to the equivalent counts by using the FPM\_InitFixedPoint\_m() macro within the #define statement.

#### Local

|  |  |  |  |
| --- | --- | --- | --- |
| Constant Name | Resolution | Units | Value |
| D\_OTP1BITERRADDR\_CNT\_U32 | 1 | Counts | 0xF00803F4 |
| D\_OTP2BITERRADDR\_CNT\_U32 | 1 | Counts | 0xF00803FC |
| D\_DPRAMSELECT\_CNT\_U32 | 1 | Counts | 0x4 |
| D\_SPRAMSELECT\_CNT\_U32 | 1 | Counts | 0x8 |
| D\_EFCAUTOLOADERROREN\_CNT\_U32 | 1 | Counts | 0x00040000 |
| D\_EFCINSTRUCTIONERROREN\_CNT\_U32 | 1 | Counts | 0x00080000 |
| D\_EFCINSTRUCTIONINFOEN\_CNT\_U32 | 1 | Counts | 0x00100000 |
| D\_EFCSELFTESTERROREN\_CNT\_U32 | 1 | Counts | 0x00200000 |
| D\_EFCSELFTESTDONE\_CNT\_U32 | 1 | Counts | 0x00008000 |
| D\_EFCSELFTESTERROR\_CNT\_U32 | 1 | Counts | 0x00004000 |
| D\_OUTPUTENABLE\_CNT\_U32 | 1 | Counts | 0x0003C000 |
| D\_SELFTESTERROR\_CNT\_U32 | 1 | Counts | 0x18 |
| D\_LPOTRIMVALUE\_CNT\_U32 | 1 | Counts | ((\*(uint32\*)0xF00801B4) & 0xFFFF0000)>>16 |
| D\_PLLSLIPMASK\_CNT\_U32 | 1 | Counts | 0x00000300 |
|  |  |  |  |
| D\_LOWBYTEMASK\_CNT\_U32 | 1 | Counts | 0x0000000FFUL |
| D\_CURRENTLPOTRIMMASK\_CNT\_U32 | 1 | Counts | 0x00000FFFFUL |
| D\_LPOMONLFTRIMMASK\_CNT\_U32 | 1 | Counts | 0xFFFFFF00UL |
| D\_LPOMONHFTRIMMASK\_CNT\_U32 | 1 | Counts | 0xFFFF00FFUL |
| D\_OTPRAMDEVICEADDR\_CNT\_U32 | 1 | Counts | 0xF0080148u |
| D\_ESRAM5ENABLE\_CNT\_U32 | 1 | Counts | 0x01<<20 |
| D\_ESRAM6ENABLE\_CNT\_U32 | 1 | Counts | 0x01<<21 |
| D\_ESRAM8ENABLE\_CNT\_U32 | 1 | Counts | 0x01<<27 |

#### Global

This section lists the global constants used by the module. For details on global constants, refer to the Data Dictionary for the application.

|  |
| --- |
| Constant Name |
| D\_SPRAMGRPSTMS\_CNT\_U32 |
| D\_DPRAMGRPSTMS\_CNT\_U32 |
| D\_CSDISCLRVAL\_CNT\_U32 |
| D\_CSVSTATMASK\_CNT\_U32 |
| D\_PCSPWRDWNCLR0VAL\_CNT\_U32 |
| D\_PCSPWRDWNCLR1VAL\_CNT\_U32 |
| D\_PSPWRDWNCLR0VAL\_CNT\_U32 |
| D\_PSPWRDWNCLR1VAL\_CNT\_U32 |
| D\_PSPWRDWNCLR2VAL\_CNT\_U32 |
| D\_PSPWRDWNCLR3VAL\_CNT\_U32 |
| D\_RAMPWRONINITMASKTMS\_CNT\_U32 |
| D\_RAMRESETINITMASKTMS\_CNT\_U32 |
| D\_PLLCTL1VAL\_CNT\_U32 |
| D\_FRDCNTLVAL\_CNT\_U32 |

### Module specific Lookup Tables Constants

(This is for lookup tables (arrays) with fixed values, same name as other tables)

|  |  |  |  |
| --- | --- | --- | --- |
| Constant Name | Resolution | Value | Software Segment |
| None |  |  |  |

# Functions/Macros used by the Sub-Modules

## Library Functions / Macros

The library and functions / Macros that are called by the various sub modules are identified below,

1. <None>

## Data Hiding Functions

1. <None>

## Global Functions/Macros Defined by this Module

### Global Function #1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | (Exact name used) | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | (if none, write None) |  |  |  |  |
|  | (Insert more rows for additional passed arguments) |  |  |  |  |
| **Return Value** | (if no value returned, write N/A) |  |  |  |  |

#### Description

(Place flowchart/design for local function)

## Local Functions/Macros Used by this MDD only

### Local Function DiagFailedReset

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | DiagFailedReset | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Description



### Local Function resetStartup

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | resetStartup | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | *MemInitMask\_Cnt\_T\_u32* | uint32 | FULL | FULL |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Description



### Local Function pwronStartup

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | pwronStartup | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Description



### Local Function afterSTC

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | afterSTC | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Description



### Local Function memInitialization

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | memInitialization | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | *MemInitMask\_Cnt\_T\_u32* | uint32 | FULL | FULL |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Design Rationale

All VIM registers are cleared at the end of the startup initialization routine because the startup routine tests will set some interrupt flags during the testing (CCMSelfCheck).

#### Description



### Local Function cpuSelfTest

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | cpuSelfTest | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Description



### Local Function setupPLL

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | setupPLL | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Design Rationale

This function configures the PLL settings. These register settings assume 20MHz oscillator. Changes to these values will drive changes to these register values.

#### Description



### Local Function efcCheck

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | efcCheck | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Description



### Local Function efcStuckZeroTestPassed

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | efcStuckZeroTestPassed | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | *TestPassed\_Cnt\_T\_lgc* | boolean | FALSE | TRUE | 0 |

#### Description



### Local Function efcSelfTest

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | efcSelfTest | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Description



### Local Function periphInit

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | periphInit | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Description



### Local Function checkEFCSelfTestPassed

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | checkEFCSelfTestPassed | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | *TestPassed\_Cnt\_T\_lgc* | boolean | FALSE | TRUE | 0 |

#### Description



### Local Function setupFlash

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | setupFlash | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Design Rationale

The device ID is checked and if the gladiator revA part ID is detected, pre-emption is disabled. This is to account for errata DEVICE#145 of revA parts.

#### Description



### Local Function trimLPO

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | trimLPO | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Description



### Local Function fmcBus2Check

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | fmcBus2Check | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Description



### Local Function fmcECCcheck

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | fmcECCcheck | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Description



### Local Function mapClocks

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | mapClocks | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Description



### Local Function stcSelfCheck

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | stcSelfCheck | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Design Rationale

This function assumes a divide by two on the clock frequency will produce a frequency for the test to run at < 90MHz. For the current design, the clock frequency is 160MHz which will produce an 80MHz clock for this test.

#### Description



### Local Function ccmSelfCheck

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | ccmSelfCheck | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Description



### Local Function pbistSelfCheck

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | pbistSelfCheck | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Design Rationale

This function assumes a divide by two on the clock frequency will produce a frequency for the test to run at < 90MHz. For the current design, the clock frequency is 160MHz which will produce an 80MHz clock for this test.

#### Description



### Local Function pbistRun

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | pbistRun | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | *raminfoL\_Cnt\_T\_u32* | uint32 | FULL | FULL |  |
|  | *algomask\_Cnt\_T\_u32* | uint32 | FULL | FULL |  |
| **Return Value** | N/A |  |  |  |  |

#### Design Rationale

This function assumes a divide by two on the clock frequency will produce a frequency for the test to run at < 90MHz. For the current design, the clock frequency is 160MHz which will produce an 80MHz clock for this test.

#### Description



### Local Function pbistStop

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | pbistStop | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Description



### Local Function Delay

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | Delay | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** | N/A |  |  |  |  |

#### Description



### Local Macro GetRamDomian1Enabled

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | GetRamDomian1Enabled | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** |  | boolean | FALSE | TRUE | 1 |

#### Description



### Local Macro GetRamDomian2Enabled

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | GetRamDomian2Enabled | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** |  | boolean | FALSE | TRUE | 1 |

#### Description



### Local Macro GetRamDomian3Enabled

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | GetRamDomian3Enabled | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None |  |  |  |  |
|  |  |  |  |  |  |
| **Return Value** |  | boolean | FALSE | TRUE | 1 |

#### Description



# Software Module Implementation

## Runtime Environment (RTE) Initial Values

This section lists the initial values of data written by this module but controlled by the RTE. After RTE initialization, the data in this table will contain these values.

|  |  |
| --- | --- |
| Data | Value |
| <None> |  |

## Initialization Functions

(Note: For multiple init functions, insert new headers at the “Header 2” level – subset of “5.1 Initialization Functions” and follow the same sub-section design shown below)

### Init: \_c\_int00 / Startup

#### Design Rationale

This \_c\_int00 function is designed to be placed at the reset vector. Since this function needs to be compiled in arm mode, it immediately branches to the Startup() function which can optionally be compiled in thumb mode to allow memory optimizations. Since stacks are not available, the Startup() function needs to be defined as a Reset vector similar to \_c\_int00 so that the compiler does not try pushing data to the stack on entry to this function.

##### TI Recommended Initialization

Texas Instruments had provided a recommended startup sequence which the design was based from. The document (spna106a.pdf) and accompanying source code (spna106a.zip) are located in the doc folder of this SWC. Note that SysStartup code primarily handles up to step 35 of spna106a (excluding step 12, 29, 30). Some of the notable deviations from the document include:

* Addition of debugger connection detection to allow skipping initialization steps which interfere with debugging
* Replacement of while(1) loops where failures occurred with Nexteer’s failure strategy summarized in section below
* Use of configuration template headers to define initialization constants that may change from program to program
* Handling of resets and use of “ResetCause” variable
* Moving of mux initialization (Step 12 in spna106a) into application specific code (out of common initialization sequence)
* Moving of SECDED logic check on RAM and Flash (Step 29, 30) into application specific code (out of common initialization sequence)
* Some of the functions names were changed and the return type was changed to add clarity to the design (efcStuckZeroTest, checkEFCSelfTest)

##### Failed Initialization Diagnostic Strategy

A common strategy is used when diagnostics that are part of this initialization routine fail. In general, the “ResetCause” variable is set to an appropriate value, the nError pin is forced low, and a software reset is performed. The nError will put the controller into a known safe state. The software reset will allow the rest of the initialization tests to be bypassed, as well as reset the controller’s registers to a known state. The rest of the tests are bypassed to ensure the first failure is captured as the “ResetCause”, and the application can then evaluate the failure reason and set appropriate diagnostics codes (and communicate these on the communication bus if desired). Note that the registers getting reset will also reset the nError pin state, so forcing the nError pin low is also done when processing the software reset. Also note that for any of these diagnostics that fail, the general RAM banks will be initialized as part of the reset handling (as opposed to just initializing peripheral ram like most other resets).

##### Software Initiated Resets

For the two types of resets which can be initiated by software (CPU reset and SW reset), a strategy for setting the “ResetCause” variable has been used to allow for flexibility in adding new reset causes. A check is done to see if the current value of the “ResetCause” indicates a power-on reset. If this is the case, the initialization code assumes whatever code that initiated the reset did not explicitly indicate the reason for reset, and therefore the initialization code overwrites the value to the appropriate reset type (CPU or SW). If the reset value is anything other than power-on reset, the initialization code leaves the “ResetCause” as-is under the assumption that whatever code that initiated the reset set this cause specifically.

##### Errata Processing

The device ID is checked at power-on and if the gladiator revA part ID is detected, a \_esmCcmErrorsClear\_() function is called to handle DEVICE#140 errata processing.

#### Processing



## Periodic Functions

None

## Fault Recovery Functions

None

## Shutdown Functions

None

## Interrupt Functions

None

## Serial Communication Functions

None

# Execution Requirements

## Execution Sequence of the Module

## Execution Rates for sub-modules called by the Scheduler

This table serves as reference for the Scheduler design

|  |  |  |
| --- | --- | --- |
| Function Name | Calling Frequency | System State(s) in which the function is called |
| \_c\_int00() | power on and during a reset | N/A |

## Execution Requirements for Serial Communication Functions

|  |  |
| --- | --- |
| Function Name | Sub-Module called by (Serial Comm Function Name) |
| <None> |  |

# Memory Map Definition Requirements

## Sub Modules (Functions)

This table identifies the software segments for functions identified in this module.

|  |  |
| --- | --- |
| Name of Sub Module | Software Segment |
| \_c\_int00() |  |

## Local Functions

This table identifies the software segments for local functions identified in this module.

|  |  |
| --- | --- |
| Name of Sub Module | Software Segment |
| Startup |  |
| DiagFailedReset | inline with calling function |
| resetStartup |  |
| pwronStartup | inline with calling function |
| afterSTC | inline with calling function |
| setupPLL | inline with calling function |
| trimLPO | inline with calling function |
| setupFlash | inline with calling function |
| periphInit | inline with calling function |
| mapClocks | inline with calling function |
| ccmSelfCheck | inline with calling function |
| stcSelfCheck | inline with calling function |
| cpuSelfTest | inline with calling function |
| efcCheck | inline with calling function |
| efcSelfTest | inline with calling function |
| efcStuckZeroTestPassed | inline with calling function |
| checkEFCSelfTestPassed | inline with calling function |
| fmcBus2Check | inline with calling function |
| fmcECCcheck | inline with calling function |
| pbistSelfCheck | inline with calling function |
| pbistRun | inline with calling function |
| pbistStop | inline with calling function |
| memInitialization | inline with calling function |

# Known Issues / Limitations With Design

1. (Item #1)

# Revision Control Log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item #** | **Rev #** | **Change Description** | **Date** | **Author Initials** |
| 1 | 1 | Initial creation | 05/11/12 | LWW |
| 2 | 2 | Updates for LPO Trim issue | 06/11/12 | LWW |
| 3 | 3 | Updates to turn of PLL Slip and clock monitor resets in PLLCTL1 Register, added clearing of VIM interrupt requests before jumping into bootloader, added turning on DCAN parity if the memory is configured to be powered on prior to the memory initialization call | 07/13/12 | LWW |
| 4 | 4 | Update SetUpPll code to check and recover from PLLSLIP and/or FBLSLIP. Modified SetUpPll code, Updated the TrimLPO function and added delay() function | 09/14/12 | BG, BDO |
| 5 | 5 | Updated PLL settings for clock change to 150MHz | 10/12/12 | BWL |
| 6 | 6 | Made PLL configurable and tied flash waitstate settings to frequency | 02/14/13 | LWW |
| 7 | 7 | Made RAM pbist only run on banks enabled in OTP memory | 02/15/13 | BWL |
| 8 | 8 | Correction for static register check | 02/15/13 | BWL |
| 9 | 9 | Added call to Startup to allow compilation of file in thumb mode | 08/02/13 | LWW |