

PRELIMINARY

|  |  |
| --- | --- |
| CEC1702 OTP Programming Design Doc | |
| Specification | |
| Rev 4 | 02/17/2017 |

MICROCHIP CONFIDENTIAL

|  |
| --- |
| Copyright © 2012 Microchip or its subsidiaries. All rights reserved.  The information contained herein is confidential and proprietary to Microchip, shall be used solely in accordance with the agreement pursuant to which it is provided, and shall not be reproduced or disclosed to others without the prior written consent of Microchip. Although the information is believed to be accurate, no responsibility is assumed for inaccuracies. Microchip reserves the right to make changes to this document and to specifications and product descriptions at any time without notice. Neither the provision of this information nor the sale of the described semiconductor devices conveys any licenses under any patent rights or other intellectual property rights of Microchip or others. The product may contain design defects or errors known as anomalies, including but not necessarily limited to any which may be identified in this document, which may cause the product to deviate from published specifications. Microchip products are not designed, intended, authorized or warranted for use in any life support or other application where product failure could cause or contribute to personal injury or severe property damage. Any and all such uses without prior written approval of an officer of Microchip will be fully at the risk of the customer.    MICROCHIP DISCLAIMS AND EXCLUDES ANY AND ALL WARRANTIES, INCLUDING WITHOUT LIMITATION ANY AND ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, AND AGAINST INFRINGEMENT AND THE LIKE, AND ANY AND ALL WARRANTIES ARISING FROM ANY COURSE OF DEALING OR USAGE OF TRADE. IN NO EVENT SHALL SMSC BE LIABLE FOR ANY DIRECT, INCIDENTAL, INDIRECT, SPECIAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES; OR FOR LOST DATA, PROFITS, SAVINGS OR REVENUES OF ANY KIND; REGARDLESS OF THE FORM OF ACTION, WHETHER BASED ON CONTRACT; TORT; NEGLIGENCE OF SMSC OR OTHERS; STRICT LIABILITY; BREACH OF WARRANTY; OR OTHERWISE; WHETHER OR NOT ANY REMEDY OF BUYER IS HELD TO HAVE FAILED OF ITS ESSENTIAL PURPOSE, AND WHETHER OR NOT SMSC HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. |

Table of Contents

[1 Introduction 3](#_Toc475120627)

[1.1 Purpose 3](#_Toc475120628)

[1.2 References 3](#_Toc475120629)

[1.3 Deliverables 3](#_Toc475120630)

[1.4 Assumptions and Dependencies 3](#_Toc475120631)

[2 CEC1702 Development Environment 4](#_Toc475120632)

[2.1 Hardware Requirements 4](#_Toc475120633)

[2.1.1 MCHP CEC1702 Blank parts 4](#_Toc475120634)

[2.1.2 JTAG interface 4](#_Toc475120635)

[2.1.3 Debug Trace FIFO Acquirer 5](#_Toc475120636)

[2.2 Software Requirements 5](#_Toc475120637)

[2.2.1 IDE 5](#_Toc475120638)

[2.2.2 User Interface tool 5](#_Toc475120639)

[2.2.3 External Tools 6](#_Toc475120640)

[3 CEC1702 Custom OTP Programming Details 7](#_Toc475120641)

[3.1.1 Firmware Programming sequence 7](#_Toc475120642)

[3.1.2 Custom OTP region targeted 7](#_Toc475120643)

[3.1.3 Error Checking 7](#_Toc475120644)

[3.1.4 Verification 7](#_Toc475120645)

[4 OTP Programming Flow 8](#_Toc475120646)

[5 Questions 9](#_Toc475120647)

[6 Appendix 10](#_Toc475120648)

[6.1 Using JTAG to download the OTP programming 10](#_Toc475120649)

[6.1.1 Ex: Using Keil JTAG 10](#_Toc475120650)

[7 Revision History 12](#_Toc475120651)

# Introduction

## Purpose

The document gives the introduction for the OTP programming design approach as well as the requirements both hardware and software for the custom OTP programming once the customer get the blank CEC1702 device

## References

Bootrom Spec

CEC1702 Data sheets

Provisioning Concept Document.doc

## Deliverables

1. Binary for SRAM download – with default to all zeros for efuse data table(PRG\_TL)
2. Custom GUI tool for efuse data generation – which have
   1. Generate EFUSE data for the fields specified (GEN\_TL)
   2. Provision for key generation for Authentication and Encryption (SSL\_TL)
   3. Generate appropriate key files for SPI image tools requirements (SSL\_TL)
   4. Append efuse table to the SRAM binary for downloading via JTAG

## Assumptions and Dependencies

1. The blank CEC1702 devices will come with ATE mode enabled by default; so JTAG will be enabled for 4 pin mode.
2. All required JTAG pins are available for wiring out to corresponding JTAG interface
3. Exact OTP programming voltage is available and routed to the appropriate pins (For Voltage regulation and pin reference Refer data sheet)
4. The functionality for OTP programming will be provided in binary format – which can be downloaded to the SRAM via JTAG and executed for the proper custom values in the efuse.
5. Proper tools will be provided for the custom efuse file generation – which include key generation and proper file generation
6. Will be using third party tools for key generation and some crypto functionality by the developed tools –openssl for key generation, Python for some crypto operation like key data extraction, file update etc..
7. Keil IDE will be used for the development and Keil Ulink pro will be used for the debug purpose
8. ECDH Key encryption for the image encryption will not be encrypted with the ROM keys

# CEC1702 Development Environment

MCHP CEC1702 development environment

**CEC1702**

***µBUS 2***

**SPI**

**JTAG**

***UART***

***I2C***

***UART***

***I2C***

**OTP**

***Tools for Provisioning***

***Generating keys and Program efuse***

***USB***

***USB***

***µBUS 1***

## Hardware Requirements

Listed the hardware requirement used for the development for the OTP programming; all the listed things below will not be part of final collateral deliverables except the required ones.

### MCHP CEC1702 Blank parts

CEC1702 parts will come with ATE mode enabled and other necessary or required fields programmed. JTAG will be enabled by default and proper voltage for OTP programming is probed.

### JTAG interface

For development purpose will use Keil ULINK pro JTAG; for binary download and executing.

### Debug Trace FIFO Acquirer

This software uses to decode the real time trace output sent from MEC14xx FW via MCHP Pegasus Trace FIFO Board. May not be part of the final collateral package.

## Software Requirements

### IDE

Keil µVision IDE will be using for the code development. The tool details

IDE-Version:

µVision V5.17.0.0

Copyright (C) 2015 ARM Ltd and ARM Germany GmbH. All rights reserved.

Tool Version Numbers:

Toolchain: MDK-ARM Standard Cortex-M

Toolchain Path: C:\Keil\_v5\ARM\ARMCC\Bin

C Compiler: Armcc.exe V5.06 update 1 (build 61)

Assembler: Armasm.exe V5.06 update 1 (build 61)

Linker/Locator: ArmLink.exe V5.06 update 1 (build 61)

Library Manager: ArmAr.exe V5.06 update 1 (build 61)

Hex Converter: FromElf.exe V5.06 update 1 (build 61)

CPU DLL: SARMCM3.DLL V5.17

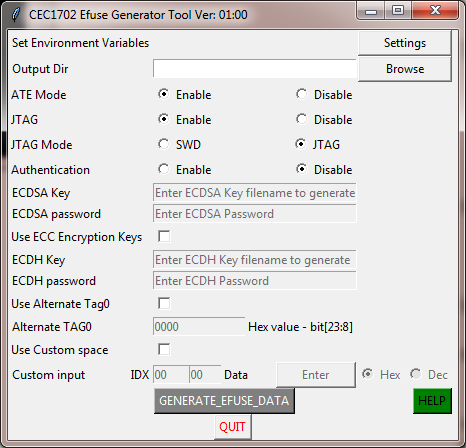
Dialog DLL: DCM.DLL V1.13.4.0

Target DLL: ULP2CM3.DLL V2.201.1.0

Dialog DLL: TCM.DLL V1.14.6.0

### User Interface tool

The user interface tool used for the OTP programming will get the inputs for the custom efuse data. Python 3.5.1 will be used for the template design



### External Tools

Used third party tool OpenSSL for the key generation – which may be used for the SPI image generation for binary image Authentication or Encryption. Listed some of the tool used

1. Python 3.5.1 – for the external interface design and for data parsing and output file generation
2. Openssl – For Key generation and some crypto functionalities
3. GO programming language – for SPI image generator and for key extractions

# CEC1702 Custom OTP Programming Details

CEC1702 device if the ATE mode is enabled will be in an infinite spin loop with the JTAG enabled in 4 pin mode. In the infinite spin loop bootrom will be checking for any address on DMA Channel 13 Memory Start Address register.

The OTP programming code will be designed to fetch data from predefined table with index data approach and program the corresponding efuse bit with the value provided.

### Firmware Programming sequence

The efuse will be programmed one bit at a time for the corresponding address offset provide for the high and low address and proper enabling of the block for the OTP engine active. The data for the efuse will be in predefined table with index data approach. The Table will be generated by the external tool and appended to the binary before downloading.

### Custom OTP region targeted

Only the listed OTP bit will be targeted to program during the execution of the firmware.

1. ATE mode disable bit - Efuse byte 35 bit[7]
2. JTAG enable bit on Bootrom Exit – Efuse byte 34 bit[1] – Bit[4]
3. Authentication Bit for the ECDSA key generation – Efuse byte 483 bit[0]
4. Authentication key region – efuse bytes 128 – 191
5. ECC private key region – Efuse byte 0 - 31
6. Custom User space region – efuse byte 192 - 479
7. Custom TAG update region – efuse bytes 508 -509

### Error Checking

Proper error checking or restriction will be added for not writing to any unknown or privileged access bits in the efuse for safety purpose. Error condition will be added in tools as well as in the code only bits defined in the section 3.1.2 will be targeted for programming.

### Verification

Only programmed custom OPT fields will be read back for proper efuse update for the provided data from. No other fields are checked for any data validation.

# OTP Programming Flow

The custom OTP programming sequences as follows

1. Customer will generate the required efuse data from the custom GUI tool
2. Custom Tool will merge the generated efuse data with the provided binary
3. Proper OTP programming voltage need to be provided to the specific pins
4. Download the merged binary to the specified region in SRAM memory via JTAG
5. Halt the processor and write the PC to the reset handler address of the provided binary or write the DMA address register with the reset handler address.
6. Release the halt instruction and the downloaded binary will be executed for the proper OTP values programmed to the specified fields.

# Questions

1. For verification of data written – Plan is to verify only the custom programmed fields not all the efuse
2. For Engineering samples – GEN\_TL to generate the efuse Data described in Provisioning Concept Document.doc, Question is whether the **efuse data** refer to only custom programmable data bits or full efuse data including Microchip Area?
3. For Production parts – described in Provisioning Concept Document.doc for generating the ATE hex file from customer information – do the tool need to merge the Customer data fields and Microchip efuse data area?
   1. If yes – Need the efuse data for Microchip area and need to be part of the tool package and will be different for different customers.
4. If customer is generating multiple efuse skews – care should be taken for handling and safe keeping the generated key PEM which is used for successful image creation from the SPI image generator (KEY PEM files are input to the SPI image generator)

# Appendix

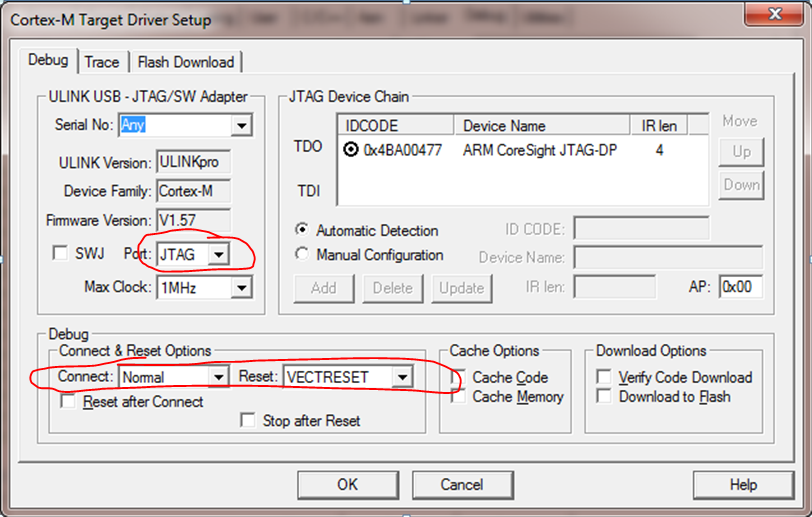
## Using JTAG to download the OTP programming

Once the efuse data is generated with the tool provided, the output OTP programming binary is combined with the efuse data using any JTAG interface we can download the binary to the target memory location 0xE0000 and set the PC address to the Reset address vector which is DWORD @ offset 4 of the provided binary.

### Ex: Using Keil JTAG

Once the IDE is opened for any dummy project with the following Options for the Taget 🡪 Debug 🡪 ULINK pro debugger 🡪 Settings as

Port as JTAG - Connect Normal under VECTRESET



Once the debugger in launched Keil IDE 🡪 Debug 🡪 Start/Stop Debug section



Use command instruction to load the generated binary and set PC = Reset handler address

**Load CEC1702\_efuse\_prog.HEX**

**eval pc= 0xE034D**

**or**

We can have the steps in an initialization file which can be added to the project settings and once the debug is started the file will be executed to load the hex file and set PC @ the reset handler address and start executing.

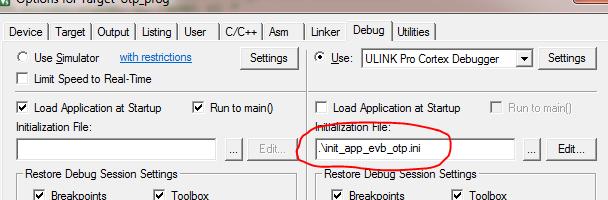
Save the following as a \*.ini file

//Initialization file for the application code

**Load CEC1702\_efuse\_prog.HEX**

**eval PC = 0x000E034D**

**g**



**Alternate way to set the PC address**

Upon reset Bootrom will be in ATE mode enabling JTAG and in a spin loop checking for any valid entry point @ DMA channel address 0x4000\_2738. So after loading the hex file one can set the value @ address 0x4000\_2738 to the reset handler address itself to launch the downloaded code.

**eval \*(unsigned int\*)0x40002738 = 0xE034D**

# Revision History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Revision Level | Date | Section | Remarks |
| Arun Krishnan - C21798 | 1.0 | 30 Jan 2017 | All section | Initial draft |
| Arun Krishnan - C21798 | 2.0 | 31 Jan 2017 | 1.3 Deliverables  3.1.4. Verification  5 Questions | Updated the names matching with the Provisioning Concept Document.doc |
| Arun Krishnan - C21798 | 3.0 | 07 Feb 2017 | Added Appendix section | Added Keil programming vis JTAG for the OTP Programming |
| Arun Krishnan - C21798 | 4.0 | 17 Feb 2017 | 2.2.2 User Interface tool | Updated the snap shot for the user interface tool |