**EQEP**

**Software Design Document**

July 2020

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Description of Versions / Changes** | **Responsible Party** | **Date** |
| 0.01 | Initial version | Gaviraju D B | July 10, 2020 |

Contents

[1 Introduction 4](#_Toc45788793)

[1.1 Functional Overview 4](#_Toc45788794)

[1.2 Assumptions and Constraints 4](#_Toc45788795)

[1.3 Relationship to Other Architecture Documents 4](#_Toc45788796)

[1.4 Stakeholders 4](#_Toc45788797)

[1.5 Notational Conventions 4](#_Toc45788798)

[Notes 4](#_Toc45788799)

[1.6 Glossary 4](#_Toc45788800)

[1.7 References 4](#_Toc45788801)

[1.8 Features Not Supported 5](#_Toc45788802)

[2 Design Description 5](#_Toc45788803)

[2.1 Functional/Logical Viewpoint 5](#_Toc45788804)

[Overview 5](#_Toc45788805)

[Directory Structure 5](#_Toc45788806)

[Component Interaction 6](#_Toc45788807)

[Interfaces 6](#_Toc45788808)

[2.2 Process/Concurrency Viewpoint 6](#_Toc45788809)

[2.3 Resource Definitions 6](#_Toc45788810)

[2.4 Interrupt Service Routines 7](#_Toc45788811)

[2.5 Error Handling 7](#_Toc45788812)

[3 Risks 7](#_Toc45788813)

[4 Requirements Traceability 7](#_Toc45788814)

[5 API Definitions 7](#_Toc45788815)

[5.1 Interfaces 7](#_Toc45788816)

[Configure the interrupt domain 7](#_Toc45788817)

[Configure the timer domain 8](#_Toc45788818)

[Configure the eqep status 8](#_Toc45788819)

[Read back of written registers 8](#_Toc45788820)

[Configure the module to get an speed & position 8](#_Toc45788821)

[6 Directory 9](#_Toc45788822)

[6.1 Index 9](#_Toc45788823)

[6.2 Glossary 9](#_Toc45788824)

[6.3 Acronym List 9](#_Toc45788825)

[Template Revision 10](#_Toc45788826)

# Introduction

The enhanced quadrature encoder pulse (eye) module is used for direct interface with a linear or rotary incremental encoder to get position, direction, and speed information from a rotating machine for use in a high-performance motion and position-control system.

## Functional Overview

This document will provide an overview of the software design of the enhanced quadrature encoder pulse (eel) IP.

The speed, direction, and position measurements are critical aspects of high performance motor control applications. This information is extracted from a group of encoder signals generated by optical encoders mounted on the motor. This group typically consists of two square waves and a pulse signal that can see once every motor revolution. The square waves are 90° apart, which helps detect the direction of rotation by determining which signal leads/lags the other. The third signal provides an index pulse and is identified using different terms such as index, marker, home position, and zero reference. This index signal can be used to indicate an absolute position.

## Assumptions and Constraints

None

## Relationship to Other Architecture Documents

None

## Stakeholders

|  |  |
| --- | --- |
| TI SW Developers | SW developers can refer to this design for developing diagnostic SW for EQEP |

## Notational Conventions

### Notes

None

## Glossary

See Directory.

## References

See Directory.

## Features Not Supported

N/A

# Design Description

## Functional/Logical Viewpoint

### Overview

To configure the eQEP registers, the CSL functional layer APIs are invoked. The eQEP Chip Support Library APIs are designed to provide APIs to configure the register of the eQEP IP. APIs are tested using CSL example applications

### Directory Structure

The eQEP CSL functional layer implementation would be implemented with below directory structure.

<csl>  
 ├── cslr\_eqep.h  
 ├── csl\_eqep.h  
 ├──src/ip/eqep/  
 ├── src\_files\_eqep.mk  
 ├── V0\_1  
 │   └── cslr\_eqep.h  
 └── V0\_2  
 ├── cslr\_eqep.h  
 ├── csl\_eqep.h  
 └── priv  
 ├── csl\_eqep.c

|  |  |
| --- | --- |
| **File Name** | **Description** |
| cslr\_eqep.h | Top level eqep register layer interface file |
| csl\_eqep.h | Top level eqep API interface include h file |
| src/ip/eqep/src\_files\_eqep.mk | Makefile for eqep build |
| src/ip/eqep/V0\_2/cslr\_eqep.h | EQEP Register Layer - V0\_2 version for AM64x |
| src/ip/eqep/V0\_2/csl\_eqep.h | EQEP Function Layer - V0\_2 version for AM64x |
| src/ip/eqep/V0\_2/priv/csl\_eqep.c | EQEP Function Layer APIs - V0\_2 version for AM64x |

### Component Interaction

The Chip Support Library (CSL) eqep APIs provide access to program the SOC eqep configuration registers.

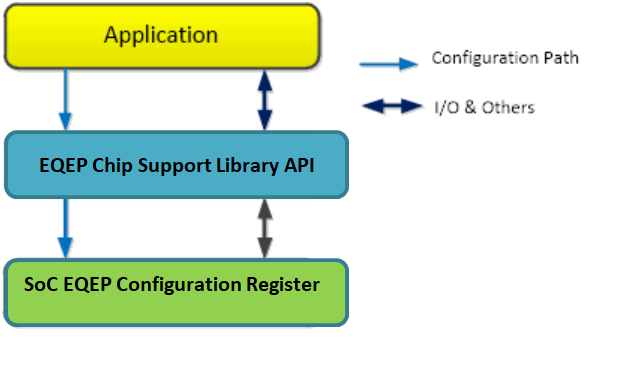


Fig. 2.1: :EQEP Application call flow diagram

### Interfaces

Design Id: (did\_csl\_eqep\_interfaces)  
Requirement: REQ\_TAG(PDK-5487)

The interfaces that are not designed to program the eQEP registers would validate the arguments for out-of-bound conditions and would also check for “NULL” pointers before programming the eQEP registers & these condition should be taken care in the application. These APIs are written at design time and the final API may be different. See the Software API Guide for the released API.

Please refer to the API definitions section.

## Process/Concurrency Viewpoint

eQEP CSL APIs are independent, memory less and stateless implementation. The CSL APIs are implemented for a single task, single thread use cases.

## Resource Definitions

The design has below footprints.

|  |  |
| --- | --- |
| **Parameter** | **Size** |
| Constant | 0 |
| BSS | 0 |
| Static | 0 |
| Data | 0 |
| Stack | < 100 |
| Code | < 100K |

## Interrupt Service Routines

The ISR routines are needed to be registered from higher layer to handle the eQEP interrupts. This would be demonstrated as part of the eQEP example code

## Error Handling

The CSL APIs is not check for NULL pointers and out of range arguments and this should be taken take in the application.

# Risks

None

# Requirements Traceability

N/A

# API Definitions

## Interfaces

The interfaces for eQEP are defined as below.

Please refer to eQEP API doxygen details for below:

1. EQEP API doxygen that describes the details on the EQEP interface API.

### Configure the interrupt domain

**void EQEP\_enableInterrupt**

**(uint32\_t baseAddr, uint16\_t intFlags)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| intFlags | uint16\_t | [IN] intFlags is a bit mask of the interrupt sources to be enabled |

This function enables individual EQEP module interrupt sources

**void EQEP\_disableInterrupt**

**(uint32\_t baseAddr, uint16\_t intFlags)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| intFlags | uint16\_t | [IN] intFlags is a bit mask of the interrupt sources to be disabled |

This function disables individual eQEP module interrupt sources

**void EQEP\_clearInterruptStatus**

**(uint32\_t baseAddr, uint16\_t intFlags)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| intFlags | uint16\_t | [IN] intFlags is a bit mask of the interrupt sources to be cleared |

This function clears eQEP module interrupt flags.

**void EQEP\_forceInterrupt**

**(uint32\_t baseAddr, uint16\_t intFlags)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| intFlags | uint16\_t | [IN] intFlags is a bit mask of the interrupt sources to be forced. |

Forces individual eQEP module interrupts.

### Configure the timer domain

**void EQEP\_enableUnitTimer**

**(uint32\_t baseAddr, uint32\_t period)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| period | Uint32\_t | [IN] period is period value at which a unit time-out interrupt is set. |

Enable the eQEP module unit timer.

**void EQEP\_disableUnitTimer**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Disable the eQEP module unit timer.

**void EQEP\_enableWatchdog**

**(uint32\_t baseAddr, uint16\_t period)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| Period | Uint32\_t | [IN] period is watchdog period value at which a time-out will occur if no quadrature-clock event is detected. |

Enable the eQEP module watchdog timer.

**void EQEP\_disableWatchdog**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Disable the eQEP module watchdog timer.

**void EQEP\_setWatchdogTimerValue**

**(uint32\_t baseAddr, uint16\_t value)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| Value | Uint16\_t | [IN] value is the value to be written to the watchdog timer. |

Set the eQEP module watchdog timer value.

### Configure the eqep status

**Bool EQEP\_isErrorSet**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Get the encoder error indicator.

This function returns the error indicator for the eQEP module. It is an error for both of the signals of the quadrature input to change at the same time.

**Returns :** true if an error has occurred otherwise false.

**void EQEP\_clearStatus**

**(uint32\_t baseAddr, uint16\_t statusFlags)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| statusFlags | Uint16\_t | [IN] statusFlags is the bit mask of the status flags to be cleared. |

Clears selected fields of the eQEP module status register.

**uint32\_t EQEP\_getStatus**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Return the content of the eQEP module status register.

**Returns:** Returnthe value of the QEP status register.

### Read back of written registers

**uint32\_t EQEP\_getCapturePeriodLatch**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Get the encoder capture period latch.

**Returns:** Return the edge-capture period latch value.

**uint32\_t EQEP\_getCapturePeriodLatch**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Get the encoder capture timer latch.

**Returns:** Return the edge-capture timer latch value.

**uint32\_t EQEP\_getPositionLatch**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Get the encoder position that was latched on a unit time-out event.

**Returns:** Return the position count latch register value.

**uint32\_t EQEP\_getStrobePositionLatch**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Get the encoder position that was latched on a strobe event.

**Returns:** Return the position count latched on a strobe event.

**uint32\_t EQEP\_getIndexPositionLatch**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Get the encoder position that was latched on an index event.

**Returns:** Return the position count latched on an index event.

**uint32\_t EQEP\_getWatchdogTimerValue**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Get the eQEP module watchdog timer value.

**Returns:** Return the current watchdog timer value.

**uint32\_t EQEP\_getCaptureTimer**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Get the encoder capture timer value.

**Returns:** Return the capture timer value.

**uint32\_t EQEP\_getCapturePeriod**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Get the encoder capture period.

**Returns:** Return the period count value between the last successive position events.

**uint32\_t EQEP\_getInterruptStatus**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Get the current interrupt status.

**Returns:** Return the current interrupt status.

**uint32\_t EQEP\_getDirection**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Get the current encoder position.

**Returns:** Return the current position of the encoder.

**uint32\_t EQEP\_getPosition**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Return the content of the eQEP module status register.

**Returns:** Return the value of the QEP status register.

### Configure the module to get an speed & position

**void EQEP\_setDecoderConfig**

**(uint32\_t baseAddr, uint16\_t config)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| Config | Uint16\_t | config is the configuration for the eQEP module decoder unit. |

Configure the eQEP module's quadrature decoder unit.

**void EQEP\_setEmulationMode**

**(uint32\_t baseAddr, eqepEmulationMode\_t emuMode)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| emuMode | eqepEmulationMode\_t | emuMode is the mode operation upon an emulation suspend. |

Set the emulation mode of the eQEP module.

**void EQEP\_setPositionCounterConfig**

**(uint32\_t baseAddr, eqepPositionResetMode\_t mode, uint32\_t maxPosition)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| Mode | eqepPositionResetMode | mode is the configuration for the eQEP module position counter. |
| maxPosition | uint32\_t | maxPosition specifies the maximum position value. |

Configure the eQEP module position counter unit.

**void EQEP\_setCaptureConfig**

**(uint32\_t baseAddr, eqepCapClkPrescale\_t capPrescale, eqeqUpEvntPrescale\_t evntPrescale);**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| capPrescale | eqepCapClkPrescale | capPrescale is the prescaler setting of the eQEP capture timer clk. |
| evntPrescale | eqeqUpEvntPrescale | evntPrescale is the prescaler setting of the unit position event frequency. |

Configure the eQEP module edge-capture unit.

**void EQEP\_setLatchMode**

**(uint32\_t baseAddr, uint32\_t latchMode)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| Latchmode | uint32\_t | latchMode is the configuration for latching of the position count and several other registers. See below for a description of this parameter. |

Configures the quadrature modes in which the position count can be latched.

**void EQEP\_enableModule**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Enable the eQEP module.

**void EQEP\_disableModule**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Disable the eQEP module.

**void EQEP\_enableCapture**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Enable the eQEP module edge-capture unit.

**void EQEP\_disableCapture**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Disable the eQEP module edge-capture unit.

**void EQEP\_enableCompare**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Enable the eQEP module position-compare unit.

**void EQEP\_disableCompare**

**(uint32\_t baseAddr)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |

Disable the eQEP module position-compare unit.

**int32\_t EQEP\_setComparePulseWidth**

**(uint32\_t baseAddr, uint16\_t cycles)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| cycles | uint16\_t | cycles is the width of the pulse that can be generated on a position-compare event. It is in units of 4 SYSCLKOUT cycles. |

Configure the position-compare unit's sync output pulse width.

**Return:** This function return CSL\_PASS if it success otherwise return CSL\_EBADARGS.

**void EQEP\_setPositionInitMode**

**(uint32\_t baseAddr, uint16\_t initMode)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| initMode | uint16\_t | initMode is the configuration for initializing the position count. |

Configures the mode in which the position counter is initialized.

**void EQEP\_setSWPositionInit**

**(uint32\_t baseAddr, Bool initialize)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| initialize | Bool | initialize is a flag to specify if software initialization of the position counter is enabled. |

Set the software initialization of the encoder position counter.

**void EQEP\_setInitialPosition**

**(uint32\_t baseAddr, uint32\_t position)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| position | uint32\_t | position is the value to be written to the position counter upon initialization. |

Set the init value for the encoder position counter.

**void EQEP\_setCompareConfig**

**(uint32\_t baseAddr, uint16\_t config, uint32\_t compareValue, uint16\_t cycles)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| config | uint16\_t | config is the configuration for the eQEP module position-compare unit. See below for a description of this parameter |
| compareValue | uint32\_t | compareValue is the value to which the position count value is compared for a position-compare event. |
| cycles | uint16\_t | cycles is the width of the pulse that can be generated on a position-compare event. It is in units of 4 SYSCLKOUT cycles. |

Configure the eQEP module position-compare unit.

**void EQEP\_setInputPolarity**

**(uint32\_t baseAddr, Bool invertQEPA, Bool invertQEPB, Bool invertIndex, Bool invertStrobe)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| invertQEPA | Bool | invertQEPA is the flag to negate the QEPA input. |
| invertQEPB | Bool | invertQEPB is the flag to negate the QEPA input. |
| invertIndex | Bool | invertIndex is the flag to negate the index input. |
| invertStrobe | Bool | invertStrobe is the flag to negate the strobe input. |

Set the polarity of the eQEP module's input signals.

**void EQEP\_setQMAModuleMode**

**(uint32\_t baseAddr, eqepQmaMode\_t qmaMode)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| qmaMode | eqepQmaMode\_t | qmaMode is the mode in which the QMA module will operate. |

Set the quadrature mode adapter (QMA) module mode.

**void EQEP\_setStrobeSource**

**(uint32\_t baseAddr, eqepStrobeSource\_t strobeSrc);**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| strobeSrc | eqepStrobeSource \_t | strobeSrc is the source of the strobe signal. |

Set the strobe input source of the eQEP module.

**void EQEP\_setPosition**

**(uint32\_t baseAddr, uint32\_t position)**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| baseAddr | uint32\_t | [IN] baseAddr is the base address of the eQEP module. |
| position | uint32\_t | position is the new position for the encoder. |

Set the current encoder position.

# Directory

## Index

## Glossary

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Term** | **Definition** | |  |  | |  |  | |  |
|  |  |
|  |  |

## Acronym List

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Acronym** | **Definition** | | SOC | System-on-Chip, an integrated circuit that incorporates many components into a single chip. | | EQEP | Enhanced quadrature encoder pulse | | CSL | Chip Support Library | |  |
|  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Template Revision | | | |  |
|  | | | |  |
| **Version** | | **Date** | **Author** | **Description** | | |
| 0.01 | | November 2017 | Jon Nafziger | Initial version | | |
| 0.02 | | July 12, 2018 | Krishna Allam | Updates to synchronize  this SDD template with the methodology de- scribed in the Software Architecture document template | | |
| 1.0 | | September 19, 2018 | Frank Fruth | Updates:   * Added a separate section/table for template revision (this table). * Cleared revision history at start of document to be reserved for document revision * Minor cosmetic changes to title page, e.g., removed literature number reference; | | |
| 1.0A | | November 19, 2018 | Sam Nelson Siluvaimani | Updates:   * Converted to RST format | | |
| 1.0B | | January 15, 2019 | Sam Nelson Siluvaimani | Updates:   * Some formatting changes and han-dling of references updated | | |