PSF Stack Configurations

Version 0.81

Table of Contents

1 Introduction

USB Power Delivery Software Framework(PSF) is software based Power Delivery Stack along with UPD350 Type-C Port Controller provides USB-PD functionality.

This documents serves as a guide to configure PSF to various PD functionalities which includes Port specific configurations (DataRole, PowerRole, PDOs, etc.,), PD features, Timers and System configurations.

1.1 References

- Microchip UPD350 Datasheet
- USB Power Delivery 3.0 Specification Revision 1.2
- USB Type-C Specification Revision 1.3

1.2 Terms and Abbreviations

Term	Definition
USB-PD	USB Power Delivery
UPD350	Microchip UPD350 Power Delivery Port Controller
Port Partner	Remote port which is connected to UPD350’s local port
Sink Role	USB Type-C Port which sinks power from its Port Partner
Source Role	USB Type-C Port which sources power to its Port Partner

2 SW License Agreement

Software License Agreement

3 Configure the stack

PSF has three user configurable files to enable user various level of configurability and porting:

PSF_Config.h -> To configure PSF to different PD features and functionality in a PSF integrated SOC platform

PSF_BoardConfig.h -> To configure the Hardware board parameters in a PSF integrated SOC platform

PSF_Port.h -> To port and integrate to any new SOC platform

This document elaborates on Configurations available with PSF_Config.h. Which contains,

- Inclusion and Exclusion of Features
- Power Delivery IDs Configuration
- System Level Configuration
- Port Specific Configuration
- Power Fault and Timers

PSF_Config.h shall be defined as per PD application and included as part of User Application Directory.

It is recommended to include this file path as one of preprocessor include path.

3.1 Configure Include/Exclude Features

This parameter is used to configure USB-PD features that is to be included or excluded in stack. User can choose to include/exclude any of the listed features based on the functional requirements. Based on this definition, part of code will be included/excluded for compilation process. This can be used effectively to reduce PSF code size if specific features aren't required.

3.1.1 INCLUDE_PD_SOURCE_ONLY

C

```
#define INCLUDE_PD_SOURCE_ONLY
```

Description

Setting the INCLUDE_PD_SOURCE_ONLY as 1, enables the stack to include the USB PD Source functionality at the compile time. User can set this define to 0 to reduce code size of the stack, if none of the PD enabled ports in the system are configured for Source operation.

Remarks

None

Example

```
#define INCLUDE_PD_SOURCE_ONLY  1(Include USB PD Source functionality
                                    in PSF Stack )

#define INCLUDE_PD_SOURCE_ONLY  0(Exclude USB PD Source functionality
```

```
                                        from PSF Stack )
```

3.1.2 INCLUDE_PD_SINK_ONLY

C

```
#define INCLUDE_PD_SINK_ONLY
```

Description

Setting the INCLUDE_PD_SINK_ONLY as 1, enables the stack to include USB PD Sink functionality at the compile time. User can set this define to 0 to reduce code size of the stack, if none of the PD enabled ports are configured for Sink operation.

Remarks

None

Example

```
#define INCLUDE_PD_SINK_ONLY    1(Include USB PD Sink functionality in
                                        PSF Stack)

#define INCLUDE_PD_SINK_ONLY    0(Exclude USB PD Sink functionality
                                        from PSF Stack)
```

3.1.3 INCLUDE_PD_3_0

C

```
#define INCLUDE_PD_3_0
```

Description

Setting the INCLUDE_PD_3_0 as 1, enables the stack to include USB Power delivery 3.0 specification features like Collision Avoidance, Extended message support via chunking along with PD 2.0 features at the compile.

User can set this define to 0 to reduce code size of the stack, if none of the PD enabled ports require PD 3.0 specific features and operates only at PD 2.0 specification.

Remarks

None

Example

```
#define INCLUDE_PD_3_0  1(Include USB PD 3.0 specific features from PSF Stack)

#define INCLUDE_PD_3_0  0(Exclude USB PD 3.0 specific features from PSF Stack)
```

3.1.4 INCLUDE_VCONN_SWAP_SUPPORT

C

```
#define INCLUDE_VCONN_SWAP_SUPPORT
```

Description

Setting the INCLUDE_VCONN_SWAP_SUPPORT as 1, enables the stack to include the VCONN Swap functionality at the compile time. User can set this define to 0 to reduce code size of the stack, if none of the PD enabled ports requires VCONN Swap functionality.

Remarks

None

Example

```
#define INCLUDE_VCONN_SWAP_SUPPORT  1(Include VCONN Swap functionality
                                        in PSF Stack )

#define INCLUDE_VCONN_SWAP_SUPPORT  0(Exclude VCONN Swap functionality
                                        from PSF Stack )
```

3.1.5 INCLUDE_POWER_FAULT_HANDLING

C

```
#define INCLUDE_POWER_FAULT_HANDLING
```

Description

Setting the INCLUDE_POWER_FAULT_HANDLING as 1, enables the stack to handle Power faults (Source & Sink OVP, Source OCS, Sink under voltage) as per Power Delivery specification Rev3.0 as applicable.

Remarks

None

Example

```
#define INCLUDE_POWER_FAULT_HANDLING    1(Include Power Fault handling to PSF Stack)

#define INCLUDE_POWER_FAULT_HANDLING    0(Exclude Power Fault handling from PSF Stack)
```

3.1.6 INCLUDE_POWER_MANAGEMENT_CTRL

C

```
#define INCLUDE_POWER_MANAGEMENT_CTRL
```

Description

Setting the INCLUDE_POWER_MANAGEMENT_CTRL as 1, enables the stack to include the functionality to put the UPD350 into low power mode if UPD350 is inactive for CONFIG_PORT_UPD_IDLE_TIMEOUT_MS (see page 23) time

Remarks

None

Example

```
#define INCLUDE_POWER_MANAGEMENT_CTRL   1(Include power management feature)

#define INCLUDE_POWER_MANAGEMENT_CTRL   0(Exclude power management feature)
```

3.1.7 INCLUDE_UPD_PIO_OVERRIDE_SUPPORT

C

```
#define INCLUDE_UPD_PIO_OVERRIDE_SUPPORT
```

Description

PIO override is UPD350 specific feature which changes the state of a PIO without SW intervention. PSF stack used this feature to disable EN_VBUS instantly on detection of a Power Fault Condition. Setting the INCLUDE_UPD_PIO_OVERRIDE_SUPPORT as 1 enables this feature.

Remarks

To use this feature, EN_VBUS and FAULT_IN Pin of the system should be UPD350 PIOs. Confined to INCLUDE_POWER_FAULT_HANDLING (see page 5) define. INCLUDE_POWER_FAULT_HANDLING (see page 5) should be declared as 1 for INCLUDE_UPD_PIO_OVERRIDE_SUPPORT define to be effective.

Example

```
#define INCLUDE_UPD_PIO_OVERRIDE_SUPPORT    1(Include UPD350 PIO Override support for Power fault to Zeus stack)

#define INCLUDE_UPD_PIO_OVERRIDE_SUPPORT    0(Exclude UPD350 PIO Override support for Power fault from Zeus stack)
```

3.1.8 INCLUDE_PDFU

C

```
#define INCLUDE_PDFU
```

Description

Setting the INCLUDE_PDFU as 1, includes the state machine code for PD Firmware Update feature as per USB Power Delivery FW Update Specification v1.0.

Remarks

None

Example

```
#define INCLUDE_PDFU    1(Include PDFU feature)

#define INCLUDE_PDFU    0(Exclude PDFU feature)
```

3.2 Power Delivery IDs Configuration

3.2.1 CONFIG_VENDOR_ID

C

```
#define CONFIG_VENDOR_ID
```

Description

CONFIG_VENDOR_ID field defines Vendor Identifier value. It is used by the PD Firmware Update state-machine during Enumeration phase. This information is shared with the PDFU Initiator as part of GET_FW_ID command's response.

Remarks

The user definition of this macro is mandatory when INCLUDE_PDFU (see page 6) is

1. It should always be two byte width.

Example

```
#define CONFIG_VENDOR_ID 0x0424u
```

3.2.2 CONFIG_PRODUCT_ID

C

```
#define CONFIG_PRODUCT_ID
```

Description

CONFIG_PRODUCT_ID is the Product Identifier value. It is used by the PD Firmware Update state-machine during Enumeration phase. This information is shared with the PDFU Initiator as part of GET_FW_ID command's response.

Remarks

The user definition of this macro is mandatory when INCLUDE_PDFU (see page 6) is

1. It should always be two byte width.

Example

```
#define CONFIG_PRODUCT_ID 0x301Cu
```

3.2.3 CONFIG_HWMAJOR_VERSION

C

```
#define CONFIG_HWMAJOR_VERSION
```

Description

CONFIG_HWMAJOR_VERSION defines Hardware Major Version details of the product. It is used by the PD Firmware Update state-machine during Enumeration phase. This information is shared with the PDFU Initiator as part of GET_FW_ID command's response.

Remarks

This is a 4-bit entity. (Valid values are 0 to 16). The user definition of this macro is mandatory when INCLUDE_PDFU (see page 6) is 1.

Example

```
#define CONFIG_HWMAJOR_VERSION 0x00
```

3.2.4 CONFIG_HWMINOR_VERSION

C

```
#define CONFIG_HWMINOR_VERSION
```

Description

CONFIG_HWMAJOR_VERSION (see page 7) defines Hardware Minor Version details of the product. It is used by the PD Firmware Update state-machine during Enumeration phase. This information is shared with the PDFU Initiator as part of GET_FW_ID command's response.

Remarks

This is a 4-bit entity. (Valid values are 0 to 16). The user definition of this macro is mandatory when INCLUDE_PDFU (see page 6) is 1.

Example

```
#define CONFIG_HWMINOR_VERSION 0x00
```

3.2.5 CONFIG_SILICON_VERSION

C

```
#define CONFIG_SILICON_VERSION
```

Description

CONFIG_SILICON_VERSION UPD301 Silicon Base Version. It is used by the PD Firmware Update state-machine during Enumeration phase. This information is shared with the PDFU Initiator as part of GET_FW_ID command's response.

Remarks

The user definition of this macro is mandatory when INCLUDE_PDFU (see page 6) is 1.

Example

```
#define CONFIG_SILICON_VERSION 0x01u
```

3.3 System Level configuration

3.3.1 CONFIG_PD_PORT_COUNT

C

```
#define CONFIG_PD_PORT_COUNT
```

Description

CONFIG_PD_PORT_COUNT defines the number of Power delivery enabled ports. The maximum value for CONFIG_PD_PORT_COUNT is 4.

Remarks

None

Example

```
#define CONFIG_PD_PORT_COUNT 2 (Number of PD ports enabled in PSF Stack is 2)
```

3.3.2 CONFIG_DEFINE_UPD350_HW_INTF_SEL

C

```
#define CONFIG_DEFINE_UPD350_HW_INTF_SEL
```

Description

CONFIG_DEFINE_UPD350_HW_INTF_SEL defines the Hardware interface for communication between the SOC and UPD350.

It can take either CONFIG_UPD350_SPI or CONFIG_UPD350_I2C as value.

CONFIG_UPD350_SPI - SPI is the communication interface between SOC and UPD350. SPI interface is supported by UPD350 B and D parts alone.

CONFIG_UPD350_I2C - I2C is the communication interface between SOC and UPD350. I2C interface is supported by UPD350 A and C parts alone.

Remarks

None.

Example

```
#define CONFIG_DEFINE_UPD350_HW_INTF_SEL    CONFIG_UPD350_SPI
#define CONFIG_DEFINE_UPD350_HW_INTF_SEL    CONFIG_UPD350_I2C
```

3.4 Port Specific Configuration

3.4.1 CONFIG_PORT_n_DATA_ROLE

C

```
#define CONFIG_PORT_n_DATA_ROLE
```

Description

CONFIG_PORT_n_DATA_ROLE refers to the data role of nth port. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8)-1). Setting CONFIG_PORT_n_DATA_ROLE as 1, configures the nth port data role as DFP or Setting CONFIG_PORT_n_DATA_ROLE as 0, configures the nth port data role as UFP. UFP and 0 as DFP

Remarks

None

Example

```
#define CONFIG_PORT_0_DATA_ROLE  1 (Configuring the Port 0 as DFP)

#define CONFIG_PORT_0_DATA_ROLE  0 (Configuring the Port 0 as UFP)
```

3.4.2 CONFIG_PORT_n_POWER_ROLE

C

```
#define CONFIG_PORT_n_POWER_ROLE
```

Description

CONFIG_PORT_n_POWER_ROLE refers to the power role of nth port. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8)-1). Setting CONFIG_PORT_n_POWER_ROLE as 1, configures the nth port

as Source or Setting CONFIG_PORT_n_POWER_ROLE as 0, configures the nth port as Sink.

Remarks

None

Example

```
#define CONFIG_PORT_0_POWER_ROLE    1 (Configuring the Port 0 as Source)

#define CONFIG_PORT_0_POWER_ROLE    0 (Configuring the Port 0 as Sink)
```

3.4.3 CONFIG_PORT_n_RP_CURRENT_VALUE

C

```
#define CONFIG_PORT_n_RP_CURRENT_VALUE
```

Description

CONFIG_PORT_n_RP_CURRENT_VALUE refers to the Rp Value of nth port. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8)-1).

Remarks

If CONFIG_PORT_n_POWER_ROLE (see page 9) set as 0 (Sink), CONFIG_PORT_n_RP_CURRENT_VALUE should be defined as 0.

Example

```
#define CONFIG_PORT_0_RP_CURRENT_VALUE   0 (Configuring the Port 0 Rp Value as Disabled)
#define CONFIG_PORT_0_RP_CURRENT_VALUE   1 (Configuring the Port 0 Rp Value as DEFAULT)
#define CONFIG_PORT_0_RP_CURRENT_VALUE   2 (Configuring the Port 0 Rp Value as CURRENT_15)
#define CONFIG_PORT_0_RP_CURRENT_VALUE   3 (Configuring the Port 0 Rp Value as CURRENT_30)
```

3.4.4 CONFIG_PORT_n_SOURCE_NUM_OF_PDOS

C

```
#define CONFIG_PORT_n_SOURCE_NUM_OF_PDOS
```

Description

CONFIG_PORT_n_SOURCE_NUM_OF_PDOS refers to the number PDOs supported by the nth source port. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8) - 1).

Remarks

CONFIG_PORT_n_SOURCE_NUM_OF_PDOS can be configured from 1 to 7.

Example

```
#define CONFIG_PORT_n_SOURCE_NUM_OF_PDOS          4
```

3.4.5 CONFIG_PORT_n_SOURCE_UNCONSTRAINED_PWR

C

```
#define CONFIG_PORT_n_SOURCE_UNCONSTRAINED_PWR
```

Description

CONFIG_PORT_n_SOURCE_UNCONSTRAINED_PWR defines the Unconstrained Power bit in PDO of nth source port. As per PD specification, this field is exposed for PDO1 alone for rest of the fixed PDOs it is Zero. CONFIG_PORT_n_SOURCE_UNCONSTRAINED_PWR can be configured as 0 or 1. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8) - 1).

Remarks

None

Example

```
#define CONFIG_PORT_n_SOURCE_UNCONSTRAINED_PWR      0
```

3.4.6 CONFIG_PORT_n_SOURCE_USB_COM

C

```
#define CONFIG_PORT_n_SOURCE_USB_COM
```

Description

CONFIG_PORT_n_SOURCE_USB_COM defines the USB communication enable bit in PDO of nth source port. As per PD specification, this field is exposed for PDO1 alone for rest of the fixed PDOs it is Zero. CONFIG_PORT_n_SOURCE_USB_COM can be configured as 0 or 1. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8) - 1).

Remarks

None

Example

```
#define CONFIG_PORT_n_SOURCE_USB_COM        1
```

3.4.7 CONFIG_PORT_n_SOURCE_USB_SUSPEND

C

```
#define CONFIG_PORT_n_SOURCE_USB_SUSPEND
```

Description

CONFIG_PORT_n_SOURCE_USB_SUSPEND defines the USB Suspend supported bit in PDO of nth source port. As per PD specification, this field is exposed for PDO1 alone for rest of the fixed PDOs it is Zero. CONFIG_PORT_n_SOURCE_PDO_1_USB_SUSPEND can be configured as 0 or 1. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8) - 1).

Remarks

None

Example

```
#define CONFIG_PORT_n_SOURCE_USB_SUSPEND        0
```

3.4.8 CONFIG_PORT_n_SOURCE_PDO_x_CURRENT

C

```
#define CONFIG_PORT_n_SOURCE_PDO_x_CURRENT
```

Description

CONFIG_PORT_n_SOURCE_PDO_x_CURRENT defines the maximum current value in xth PDO of nth source port. As per PD specification there can be 7 PDOs, So x takes value from 1 to 7. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8) - 1).

Remarks

Units are in mA

Example

```
#define CONFIG_PORT_0_SOURCE_PDO_1_CURRENT      3000 (Maximum current value is configured
as 3A for PDO1 of Port-0)
```

3.4.9 CONFIG_PORT_n_SOURCE_PDO_x_VOLTAGE

C

```
#define CONFIG_PORT_n_SOURCE_PDO_x_VOLTAGE
```

Description

CONFIG_PORT_n_SOURCE_PDO_1_VOLTAGE defines the voltage supported in xth PDO of nth source port. As per PD specification there can be 7 PDOs, So x takes value from 1 to 7. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8) - 1).

Remarks

Units are in mV. It is mandatory to define PDO1 as vSafe5V.

Example

```
#define CONFIG_PORT_0_SOURCE_PDO_1_VOLTAGE      5000 (Voltage supported is configured as 5V
for PDO1 of Port-0)
```

3.4.10 CONFIG_PORT_n_SINK_NUM_OF_PDOS

C

```
#define CONFIG_PORT_n_SINK_NUM_OF_PDOS
```

Description

CONFIG_PORT_n_SINK_NUM_OF_PDOS defines the number PDOs supported by the nth sink port. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8) - 1).

Remarks

CONFIG_PORT_n_SINK_NUM_OF_PDOS can be configured from 1 to 7.

Example

```
#define CONFIG_PORT_n_SINK_NUM_OF_PDOS      4
```

3.4.11 CONFIG_PORT_n_SINK_HIGHER_CAPABILITY

C

```
#define CONFIG_PORT_n_SINK_HIGHER_CAPABILITY
```

Description

CONFIG_PORT_0_SINK_HIGHER_CAPABILITY defines the Higher Capability bit in PDO in nth sink port. As per PD specification, this field is exposed for PDO1 alone for rest of the fixed PDOs it is Zero. CONFIG_PORT_0_SINK_HIGHER_CAPABILITY can be configured as 0 or 1. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8) - 1).

Remarks

None

Example

```
#define CONFIG_PORT_n_SINK_HIGHER_CAPABILITY        0
```

3.4.12 CONFIG_PORT_n_SINK_USB_SUSPEND

C

```
#define CONFIG_PORT_n_SINK_USB_SUSPEND
```

Description

CONFIG_PORT_n_SINK_USB_SUSPEND defines the USB Suspend supported bit in PDO in nth sink port. As per PD specification, this field is exposed for PDO1 alone for rest of the fixed PDOs it is Zero. CONFIG_PORT_n_SINK_USB_SUSPEND can be configured as 0 or 1. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8) - 1).

Remarks

None

Example

```
#define CONFIG_PORT_n_SINK_USB_SUSPEND      0
```

3.4.13 CONFIG_PORT_n_SINK_UNCONSTRAINED_PWR

C

```
#define CONFIG_PORT_n_SINK_UNCONSTRAINED_PWR
```

Description

CONFIG_PORT_n_SINK_UNCONSTRAINED_PWR defines the Unconstrained Power bit in PDO of nth sink port. As per PD specification, this field is exposed for PDO1 alone for rest of the fixed PDOs it is Zero. CONFIG_PORT_n_SINK_UNCONSTRAINED_PWR can be configured as 0 or 1. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8) - 1).

Remarks

None

Example

```
#define CONFIG_PORT_n_SINK_UNCONSTRAINED_PWR        0
```

3.4.14 CONFIG_PORT_n_SINK_USB_COM

C

```
#define CONFIG_PORT_n_SINK_USB_COM
```

Description

CONFIG_PORT_n_SINK_USB_COM defines the USB communication enable bit in PDO of nth sink port. As per PD specification, this field is exposed for PDO1 alone for rest of the fixed PDOs it is Zero. CONFIG_PORT_n_SINK_USB_COM can be configured as 0 or 1. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8) - 1).

Remarks

None

Example

```
#define CONFIG_PORT_n_SINK_USB_COM      1
```

3.4.15 CONFIG_PORT_n_SINK_PDO_x_CURRENT

C

```
#define CONFIG_PORT_n_SINK_PDO_x_CURRENT
```

Description

CONFIG_PORT_n_SINK_PDO_x_CURRENT refers to the maximum current value in xth PDO of nth sink port. As per PD specification there can be 7 PDOs, So x takes value from 1 to 7. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8) - 1).

Remarks

Units are in mA

Example

```
#define CONFIG_PORT_0_SINK_PDO_1_CURRENT        3000 (Maximum current value is configured
as 3A for PDO1 of Port-0)
```

3.4.16 CONFIG_PORT_n_SINK_PDO_x_VOLTAGE

C

```
#define CONFIG_PORT_n_SINK_PDO_x_VOLTAGE
```

Description

CONFIG_PORT_n_SINK_PDO_x_VOLTAGE defines the voltage supported in xth PDO of nth sink port. As per PD specification there can be 7 PDOs, So x takes value from 1 to 7. n can take values between 0 and (CONFIG_PD_PORT_COUNT (see page 8) - 1).

Remarks

Units are in mV. It is mandatory to define PDO1 as vSafe5V.

Example

```
#define CONFIG_PORT_0_SINK_PDO_1_VOLTAGE        5000 (Voltage supported is configured as 5V
for PDO1 of Port-0)
```

3.5 Power Fault configuration

3.5.1 CONFIG_MAX_VCONN_FAULT_COUNT

C

```
#define CONFIG_MAX_VCONN_FAULT_COUNT
```

Description

CONFIG_MAX_VCONN_POWER_FAULT_COUNT is the maximum number of back-to-back VCONN faults allowed before it disables the VCONN. A back-to-back fault is a second fault which occurs within the CONFIG_POWER_GOOD_TIMER_MS (see page 17) after the a port is automatically re-enabled from a previous fault condition. If VCONN disabled due to occurrent VCONN power fault, VCONN will be enabled only after a physical detach and re-attach.

Remarks

None.

Example

```
#define CONFIG_MAX_VCONN_FAULT_COUNT            3
```

3.5.2 CONFIG_OVER_VOLTAGE_FACTOR

C

```
#define CONFIG_OVER_VOLTAGE_FACTOR
```

Description

CONFIG_OVER_VOLTAGE_FACTOR is percentage of PDO voltage to be considered as Over Voltage for that PDO. As per PD specification desired range for fixed PDO voltage is (0.95 * PDO Voltage) to (1.05 * PDO Volatge), So CONFIG_OVER_VOLTAGE_FACTOR should be greater than the desired range.

Remarks

If 115% of the PDO voltage has to be considered as overvoltage for that PDO voltage, then define CONFIG_OVER_VOLTAGE_FACTOR as 1.15. CONFIG_OVER_VOLTAGE_FACTOR must be defined when INCLUDE_POWER_FAULT_HANDLING (see page 5) is defined.

Example

```
#define CONFIG_OVER_VOLTAGE_FACTOR          1.15
(CONFIG_PORT_0_SOURCE_PDO_1_VOLTAGE is 5000, then for PDO 1 Over voltage is 5750mV)
```

3.5.3 CONFIG_UNDER_VOLTAGE_FACTOR

C

```
#define CONFIG_UNDER_VOLTAGE_FACTOR
```

Description

CONFIG_UNDER_VOLTAGE_FACTOR is percentage of PDO voltage to be considered as under Voltage for that PDO. As per PD specification desired range for fixed PDO voltage is (0.95 * PDO Voltage) to (1.05 * PDO Volatge), So CONFIG_OVER_VOLTAGE_FACTOR (see page 15) should be less than the desired range.

Remarks

If 85% of the PDO voltage has to be considered as under voltage for that PDO voltage, then define CONFIG_UNDER_VOLTAGE_FACTOR as 0.85. CONFIG_UNDER_VOLTAGE_FACTOR must be defined when INCLUDE_POWER_FAULT_HANDLING (see page 5) is defined.

As an exceptional case this factor is not considered for VSafe5V. For Source VSafe5V, CONFIG_VSINKDISCONNECT_VOLTAGE (see page 20) is considered as Vsafe5V undervoltage instead of (CONFIG_UNDER_VOLTAGE_FACTOR * TYPEC_VBUS_5V). For Sink, VSafe5V under voltage is not applicable as when voltage is less than or equal to CONFIG_VSINKDISCONNECT_VOLTAGE (see page 20), sink becomes disconnected.

Example

```
#define CONFIG_UNDER_VOLTAGE_FACTOR         0.85
(CONFIG_PORT_0_SOURCE_PDO_2_VOLTAGE is 9000, then for PDO 1 Over voltage is 7650mV)
```

3.5.4 CONFIG_VCONN_OCS_DEBOUNCE_IN_MS

C

```
#define CONFIG_VCONN_OCS_DEBOUNCE_IN_MS
```

Description

This is macro CONFIG_VCONN_OCS_DEBOUNCE_IN_MS.

3.5.5 CONFIG_VCONN_OCS_ENABLE

C

```
#define CONFIG_VCONN_OCS_ENABLE
```

Description

CONFIG_VCONN_OCS_DEBOUNCE_IN_MS (see page 16) is debounce timer value in terms of milliseconds for VCONN overcurrent fault conditions before reacting and entering fault recovery routine.

Remarks

None.

Example

```
#define CONFIG_VCONN_OCS_DEBOUNCE_IN_MS          2
```

3.5.6 CONFIG_MAX_VBUS_POWER_FAULT_COUNT

C

```
#define CONFIG_MAX_VBUS_POWER_FAULT_COUNT
```

Description

CONFIG_MAX_VBUS_POWER_FAULT_COUNT is the maximum number of back-to-back VBUS faults allowed before shut down of the port. A back-to-back fault is a second fault which occurs within the CONFIG_POWER_GOOD_TIMER_MS (see page 17) after the a port is automatically re-enabled from a previous fault condition. During port shutdown due to occurrent fault, the device removes its CC termination and wait for port partner to get detached physically from the port to resume its normal operation.

Remarks

None.

Example

```
#define CONFIG_MAX_VBUS_POWER_FAULT_COUNT            3
```

3.5.7 CONFIG_POWER_GOOD_TIMER_MS

C

```
#define CONFIG_POWER_GOOD_TIMER_MS MILLISECONDS_TO_TICKS(0)
```

Description

After an automatic fault recovery, a CONFIG_POWER_GOOD_TIMER_MS is ran to determine whether power remains in a ?good? state for the duration of the timer, then the Fault Counter is reset. If another fault occurs before the ?Power Good Timer? expires, then the Fault Counter is incremented.

For power Source, it is the time a power source must consistently provide power without a power fault to determine the power is good and a fault condition does not exist.

For power Sink, it is the time after the sink established a contract and its consistently drawing power from VBUS without a power fault to determine that power is good and a fault condition does not exist.

Remarks

It shall be expressed in MILLISECONDS_TO_TICKS defines.

Example

```
#define CONFIG_POWER_GOOD_TIMER_MS          MILLISECONDS_TO_TICKS(10000)
```

3.6 Vsafe5V Configuration for Source and Sink

3.6.1 CONFIG_SNK_VSAFE5V_DESIRED_MAX_VOLTAGE

C

```
#define CONFIG_SNK_VSAFE5V_DESIRED_MAX_VOLTAGE
```

Description

CONFIG_SNK_VSAFE5V_DESIRED_MAX_VOLTAGE is maximum voltage acceptable for VSafe5V expressed terms of millivolts for sink. The voltage will be considered as valid Vsafe5V only if it is equal to or greater than CONFIG_SNK_VSAFE5V_DESIRED_MIN_VOLTAGE (see page 18) & less than CONFIG_SNK_VSAFE5V_DESIRED_MAX_VOLTAGE. CONFIG_OVER_VOLTAGE_FACTOR (see page 15) * 5000 will be considered as overvoltage for Vsafe5V for sink.

Valid Vsafe5V condition: CONFIG_SNK_VSAFE5V_DESIRED_MAX_VOLTAGE > Valid Vsafe5V <= CONFIG_SNK_VSAFE5V_DESIRED_MIN_VOLTAGE (see page 18)

Overvoltage condition: Vsafe5V >= CONFIG_OVER_VOLTAGE_FACTOR (see page 15) * 5000

Remarks

It is mandatory to define CONFIG_SNK_VSAFE5V_DESIRED_MAX_VOLTAGE. It must be defined in such a way that following condition is met. CONFIG_SNK_VSAFE5V_DESIRED_MAX_VOLTAGE < CONFIG_OVER_VOLTAGE_FACTOR (see page 15) * TYPEC_VBUS_5V.

Example

```
#define CONFIG_SNK_VSAFE5V_DESIRED_MAX_VOLTAGE          5500
(
```

3.6.2 CONFIG_SNK_VSAFE5V_DESIRED_MIN_VOLTAGE

C

```
#define CONFIG_SNK_VSAFE5V_DESIRED_MIN_VOLTAGE
```

Description

CONFIG_SNK_VSAFE5V_DESIRED_MIN_VOLTAGE is minimum voltage acceptable for VSafe5V expressed terms of millivolts for Sink. The voltage will be considered as valid Vsafe5V only if it is equal to or greater than CONFIG_SNK_VSAFE5V_DESIRED_MIN_VOLTAGE & less than CONFIG_SNK_VSAFE5V_DESIRED_MAX_VOLTAGE (see page 18).

Valid Vsafe5V condition: CONFIG_SNK_VSAFE5V_DESIRED_MAX_VOLTAGE (see page 18) > Valid Vsafe5V <= CONFIG_SNK_VSAFE5V_DESIRED_MIN_VOLTAGE

Remarks

It is mandatory to define CONFIG_SRC_VSAFE5V_DESIRED_MIN_VOLTAGE (see page 19). It must be defined in such a way that following condition is met. CONFIG_SNK_VSAFE5V_DESIRED_MIN_VOLTAGE > CONFIG_VSINKDISCONNECT_VOLTAGE (see page 20).

Example

```
#define CONFIG_SNK_VSAFE5V_DESIRED_MIN_VOLTAGE          4400
```

3.6.3 CONFIG_SRC_VSAFE5V_DESIRED_MAX_VOLTAGE

C

```
#define CONFIG_SRC_VSAFE5V_DESIRED_MAX_VOLTAGE
```

Description

CONFIG_SRC_VSAFE5V_DESIRED_MAX_VOLTAGE is maximum voltage acceptable for VSafe5V expressed in terms of millivolts for source. The voltage will be considered as valid Vsafe5V only if it is equal to or greater than CONFIG_SRC_VSAFE5V_DESIRED_MIN_VOLTAGE (see page 19) & less than CONFIG_SRC_VSAFE5V_DESIRED_MAX_VOLTAGE. CONFIG_OVER_VOLTAGE_FACTOR (see page 15) * 5000 will be considered as overvoltage for Vsafe5V for Source.

Valid Vsafe5V condition: CONFIG_SRC_VSAFE5V_DESIRED_MIN_VOLTAGE (see page 19) <= Valid Vsafe5V < CONFIG_SRC_VSAFE5V_DESIRED_MAX_VOLTAGE

Vsafe5V overvoltage condition: VBUS >= CONFIG_OVER_VOLTAGE_FACTOR (see page 15) * 5000

Remarks

It is mandatory to define CONFIG_SRC_VSAFE5V_DESIRED_MAX_VOLTAGE. It must be defined in such a way that following condition is met. CONFIG_SRC_VSAFE5V_DESIRED_MAX_VOLTAGE < CONFIG_OVER_VOLTAGE_FACTOR (see page 15) * TYPEC_VBUS_5V.

Example

```
#define CONFIG_SRC_VSAFE5V_DESIRED_MAX_VOLTAGE          5500
(
```

3.6.4 CONFIG_SRC_VSAFE5V_DESIRED_MIN_VOLTAGE

C

```
#define CONFIG_SRC_VSAFE5V_DESIRED_MIN_VOLTAGE
```

Description

CONFIG_SRC_VSAFE5V_DESIRED_MIN_VOLTAGE is minimum voltage acceptable for VSafe5V expressed terms of millivolts for source. The voltage will be considered as valid Vsafe5V only if it is equal to or greater than CONFIG_SRC_VSAFE5V_DESIRED_MIN_VOLTAGE & less than CONFIG_SRC_VSAFE5V_DESIRED_MAX_VOLTAGE (see page 19).

Valid Vsafe5V condition: CONFIG_SRC_VSAFE5V_DESIRED_MIN_VOLTAGE <= Valid Vsafe5V < CONFIG_SRC_VSAFE5V_DESIRED_MAX_VOLTAGE (see page 19)

Remarks

It is mandatory to define CONFIG_SRC_VSAFE5V_DESIRED_MIN_VOLTAGE. It must be defined in such a way that following condition is met. CONFIG_SRC_VSAFE5V_DESIRED_MIN_VOLTAGE > CONFIG_VSINKDISCONNECT_VOLTAGE (see page 20).

Example

```
#define CONFIG_SRC_VSAFE5V_DESIRED_MIN_VOLTAGE          4750
(
```

3.6.5 CONFIG_VSINKDISCONNECT_VOLTAGE

C

```
#define CONFIG_VSINKDISCONNECT_VOLTAGE
```

Description

CONFIG_VSINKDISCONNECT_VOLTAGE is the vSinkDisconnect mentioned in Type c specification v1.3. Specification defines it as threshold used for transition from Attached.SNK to Unattached.SNK. In PSF, CONFIG_VSINKDISCONNECT_VOLTAGE is considered as undervoltage for Vsafe5V in case of source. For Sink, if the voltage is below CONFIG_VSINKDISCONNECT_VOLTAGE, it is considered as VBUS disconnect.

Sink: If Voltage <= CONFIG_VSINKDISCONNECT_VOLTAGE -> Sink disconnected

Source: If Voltage <= CONFIG_VSINKDISCONNECT_VOLTAGE -> Source undervoltage

Remarks

It is mandatory to define CONFIG_VSINKDISCONNECT_VOLTAGE.

Example

```
#define CONFIG_VSINKDISCONNECT_VOLTAGE          3670
(
```

3.7 PDFU Configuration

3.7.1 CONFIG_RECONFIG_PHASE_WAITTIME

C

```
#define CONFIG_RECONFIG_PHASE_WAITTIME
```

Description

CONFIG_RECONFIG_PHASE_WAITTIME specifies the Wait time required for the Reconfigure state , i.e. the PDFU_Initiate request processing takes "Wait time" ms, and next request can be issued by the PDFU_Initiator after the specified wait time. This information is shared with the PDFU Initiator as part of PDFU_INITIATE command's response.

Remarks

The user definition of this macro is mandatory when INCLUDE_PDFU (see page 6) is TRUE.

Example

```
#define CONFIG_RECONFIG_PHASE_WAITTIME   0x03u //3ms wait time required
```

3.7.2 CONFIG_TRANSFER_PHASE_WAITTIME

C

```
#define CONFIG_TRANSFER_PHASE_WAITTIME
```

Description

CONFIG_TRANSFER_PHASE_WAITTIME Species the Wait time required during the Transfer state , i.e. the PDFU Data request processing takes "Wait time" ms, and next PDFU_DATA request to be issued by the initiator after the specified wait time. This information is shared with the PDFU Initiator as part of PDFU_DATA command's response.

Remarks

The user definition of this macro is mandatory when INCLUDE_PDFU (see page 6) is TRUE.

Example

```
  #define CONFIG_TRANSFER_PHASE_WAITTIME   0x03u //3ms required for processing PDFU_DATA
request
```

3.7.3 CONFIG_UPDATABLE_IMAGEBANK_INDEX

C

```
#define CONFIG_UPDATABLE_IMAGEBANK_INDEX
```

Description

CONFIG_UPDATABLE_IMAGEBANK_INDEX specifies the Image bank index for which firmware upgrade is requested (or) in other words it corresponds to the image bank index of the Updatable application as mentioned by Architecture 2 of PD FW Update Specification.

This information is used during the Reconfiguration phase to determine what application is currently executing and whether application switching to Fixed Application is required or not.

Remarks

The user definition of this macro is mandatory when INCLUDE_PDFU (see page 6) is TRUE.

Example

```
  #define CONFIG_UPDATABLE_IMAGEBANK_INDEX   0x03u
```

3.7.4 CONFIG_VALIDATION_PHASE_WAITTIME

C

```
#define CONFIG_VALIDATION_PHASE_WAITTIME
```

Description

CONFIG_VALIDATION_PHASE_WAITTIME specifies the wait time macro for the Validation state ,i.e. the PDFU_Validate command's processing takes "Wait time" ms, and next request can be issued by the Initiator after the specified wait time.

Remarks

The user definition of this macro is mandatory when INCLUDE_PDFU (see page 6) is TRUE.

Example

```
  #define CONFIG_VALIDATION_PHASE_WAITTIME   0x03u
```

3.7.5 CONFIG_PDFU_SUPPORTED

C

```
#define CONFIG_PDFU_SUPPORTED
```

Description

CONFIG_PDFU_SUPPORTED is set to 0 if firmware is not updatable during Run time. Otherwise shall be set to 1. It is used by the PD Firmware Update state-machine during Enumeration phase. This information is shared with the PDFU Initiator as part of GET_FW_ID command's response.

Remarks

The user definition of this macro is mandatory when INCLUDE_PDFU (see page 6) is TRUE.

Example

```
#define CONFIG_PDFU_SUPPORTED TRUE
```

3.7.6 CONFIG_PDFU_VIA_USBPD_SUPPORTED

C

```
#define CONFIG_PDFU_VIA_USBPD_SUPPORTED
```

Description

CONFIG_PDFU_VIA_USBPD_SUPPORTED Set to 1 to indicate support for PDFU via USB PD Firmware Update flow.Otherwise shall be set to 0. It is used by the PD Firmware Update state-machine during Enumeration phase. This information is shared with the PDFU Initiator as part of GET_FW_ID command's response.

Remarks

The user definition of this macro is mandatory when INCLUDE_PDFU (see page 6) is TRUE.

Example

```
#define CONFIG_PDFU_VIA_USBPD_SUPPORTED TRUE
```

3.7.7 CONFIG_MAX_FIRMWARE_IMAGESIZE

C

```
#define CONFIG_MAX_FIRMWARE_IMAGESIZE
```

Description

CONFIG_MAX_FIRMWARE_IMAGESIZE defines the ROM size allocated for the Updatable application. PDFU Initiator shall flash entire size during every re-flash operation. Flashing lesser or more than this Size results in error response.

Remarks

Choose Firmware Image size in such a way that integral multiple of 256. The definition of this function is mandatory when INCLUDE_PDFU (see page 6) is TRUE

Example

```
#define CONFIG_MAX_FIRMWARE_IMAGESIZE 38*1024 for 38KB Updatable application.
```

3.8 MCU Idle Timeout Configuration

3.8.1 CONFIG_PORT_UPD_IDLE_TIMEOUT_MS

C

```
#define CONFIG_PORT_UPD_IDLE_TIMEOUT_MS MILLISECONDS_TO_TICKS(15000)
```

Description

CONFIG_PORT_UPD_IDLE_TIMEOUT_MS is the idle time after which UPD350 is put to Idle by the power management control if there is no activity or interrupt in UPD350.

Remarks

It shall be expressed in MILLISECONDS_TO_TICKS define. CONFIG_PORT_UPD_IDLE_TIMEOUT_MS is valid only if INCLUDE_POWER_MANAGEMENT_CTRL (see page 5) set as 1.

Example

```
#define CONFIG_PORT_UPD_IDLE_TIMEOUT_MS    MILLISECONDS_TO_TICKS(15000)
```

3.9 Type-C and PD Specification defined Timeout configuration

PSF provides user to configure Timeouts specified by Type-C and USB Power Delivery Specification.

3.9.1 CONFIG_TYPEC_TCCDEBOUNCE_TIMEOUT_MS

C

```
#define CONFIG_TYPEC_TCCDEBOUNCE_TIMEOUT_MS MILLISECONDS_TO_TICKS(150)
```

Description

CONFIG_TYPEC_TCCDEBOUNCE_TIMEOUT_MS defines the tCCDebounce timeout specified in the USB Type C Specification. Default value of CONFIG_TYPEC_TCCDEBOUNCE_TIMEOUT_MS is set as 150 milliseconds.

Remarks

CONFIG_TYPEC_TCCDEBOUNCE_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB Type C Compliant.

Example

```
#define CONFIG_TYPEC_TCCDEBOUNCE_TIMEOUT_MS      MILLISECONDS_TO_TICKS(150)
```

3.9.2 CONFIG_TYPEC_TPDEBOUNCE_TIMEOUT_MS

C

```
#define CONFIG_TYPEC_TPDEBOUNCE_TIMEOUT_MS MILLISECONDS_TO_TICKS(10)
```

Description

CONFIG_TYPEC_TPDEBOUNCE_TIMEOUT_MS defines the tPDDebounce timeout specified in the USB Type C Specification. Default value of CONFIG_TYPEC_TCCDEBOUNCE_TIMEOUT_MS (see page 23) is set as 10 milliseconds.

Remarks

CONFIG_TYPEC_TPDEBOUNCE_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB Type C Compliant.

Example

```
#define CONFIG_TYPEC_TPDEBOUNCE_TIMEOUT_MS      MILLISECONDS_TO_TICKS(10)
```

3.9.3 CONFIG_TYPEC_VBUS_OFF_TIMER_MS

C

```
#define CONFIG_TYPEC_VBUS_OFF_TIMER_MS MILLISECONDS_TO_TICKS(650)
```

Description

CONFIG_TYPEC_VBUS_OFF_TIMER_MS defines the tVBUSOFF specified in the USB-TypeC Specification. Default value of CONFIG_TYPEC_VBUS_OFF_TIMER_MS is set as 650 milliseconds.

Remarks

CONFIG_TYPEC_VBUS_OFF_TIMER_MS can be configured depending on the microcontroller platform for the device to be USB-TypeC Compliant.

Example

```
#define CONFIG_TYPEC_VBUS_OFF_TIMER_MS                MILLISECONDS_TO_TICKS(650)
```

3.9.4 CONFIG_TYPEC_VBUS_ON_TIMER_MS

C

```
#define CONFIG_TYPEC_VBUS_ON_TIMER_MS MILLISECONDS_TO_TICKS(275)
```

Description

CONFIG_TYPEC_VBUS_ON_TIMER_MS defines the tVBUSON specified in the USB-TypeC Specification. Default value of CONFIG_TYPEC_VBUS_ON_TIMER_MS is set as 275 milliseconds.

Remarks

CONFIG_TYPEC_VBUS_ON_TIMER_MS can be configured depending on the microcontroller platform for the device to be USB-TypeC Compliant.

Example

```
#define CONFIG_TYPEC_VBUS_ON_TIMER_MS                 MILLISECONDS_TO_TICKS(275)
```

3.9.5 CONFIG_TYPEC_VCONNOFF_TIMEOUT_MS

C

```
#define CONFIG_TYPEC_VCONNOFF_TIMEOUT_MS MILLISECONDS_TO_TICKS(25)
```

Description

CONFIG_TYPEC_VCONNOFF_TIMEOUT_MS defines the tVCONNOFF specified in the USB-Type C Specification. Default value of CONFIG_TYPEC_VCONNOFF_TIMEOUT_MS is set as 25 milliseconds.

Remarks

CONFIG_TYPEC_VCONNOFF_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB-Type C.

Example

```
#define CONFIG_TYPEC_VCONNOFF_TIMEOUT_MS                MILLISECONDS_TO_TICKS(25)
```

3.9.6 CONFIG_TYPEC_VCONNON_TIMEOUT_MS

C

```
#define CONFIG_TYPEC_VCONNON_TIMEOUT_MS MILLISECONDS_TO_TICKS(10)
```

Description

CONFIG_TYPEC_VCONNON_TIMEOUT_MS defines the tVCONNON specified in the USB-Type C Specification. Default value of CONFIG_TYPEC_VCONNON_TIMEOUT_MS is set as 2 milliseconds.

Remarks

CONFIG_TYPEC_VCONNON_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB-Type C.

Example

```
#define CONFIG_TYPEC_VCONNON_TIMEOUT_MS             MILLISECONDS_TO_TICKS(10)
```

3.9.7 CONFIG_TYPEC_VCONNDISCHARGE_TIMEOUT_MS

C

```
#define CONFIG_TYPEC_VCONNDISCHARGE_TIMEOUT_MS MILLISECONDS_TO_TICKS(35)
```

Description

CONFIG_TYPEC_VCONNDISCHARGE_TIMEOUT_MS defines the tVCONNDischarge timeout specified in the USB Type C Specification. Default value of CONFIG_TYPEC_VCONNDISCHARGE_TIMEOUT_MS is set as 35 milliseconds.

Remarks

CONFIG_TYPEC_VCONNDISCHARGE_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB Type C Compliant.

Example

```
#define CONFIG_TYPEC_VCONNDISCHARGE_TIMEOUT_MS       MILLISECONDS_TO_TICKS(35)
```

3.9.8 CONFIG_TYPEC_ERRORRECOVERY_TIMEOUT_MS

C

```
#define CONFIG_TYPEC_ERRORRECOVERY_TIMEOUT_MS MILLISECONDS_TO_TICKS(500)
```

Description

CONFIG_TYPEC_ERRORRECOVERY_TIMEOUT_MS defines the tErrorRecovery timeout specified in the USB Type C Specification. Default value of CONFIG_TYPEC_ERRORRECOVERY_TIMEOUT_MS is set as 25 milliseconds.

Remarks

CONFIG_TYPEC_ERRORRECOVERY_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB Type C Compliant.

Example

```
#define CONFIG_TYPEC_ERRORRECOVERY_TIMEOUT_MS        MILLISECONDS_TO_TICKS(25)
```

3.9.9 CONFIG_PRL_CHUNKSENDERREQUEST_TIMEOUT_MS

C

```
#define CONFIG_PRL_CHUNKSENDERREQUEST_TIMEOUT_MS MILLISECONDS_TO_TICKS(26)
```

Description

CONFIG_PRL_CHUNKSENDERREQUEST_TIMEOUT_MS defines the ChunkSenderRequestTimer specified in the USB-PD Specification. Default value of CONFIG_PRL_CHUNKSENDERREQUEST_TIMEOUT_MS is set as 26 milliseconds.

Remarks

CONFIG_PRL_CHUNKSENDERREQUEST_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB PD Compliant.

Example

```
#define CONFIG_PRL_CHUNKSENDERREQUEST_TIMEOUT_MS        MILLISECONDS_TO_TICKS(26)
```

3.9.10 CONFIG_PRL_CHUNKSENDERRESPONSE_TIMEOUT_MS

C

```
#define CONFIG_PRL_CHUNKSENDERRESPONSE_TIMEOUT_MS MILLISECONDS_TO_TICKS(26)
```

Description

CONFIG_PRL_CHUNKSENDERRESPONSE_TIMEOUT_MS defines the ChunkSenderResponseTimer specified in the USB-PD Specification. Default value of CONFIG_PRL_CHUNKSENDERRESPONSE_TIMEOUT_MS is set as 26 milliseconds.

Remarks

CONFIG_PRL_CHUNKSENDERRESPONSE_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB PD Compliant.

Example

```
#define CONFIG_PRL_CHUNKSENDERRESPONSE_TIMEOUT_MS     MILLISECONDS_TO_TICKS(26)
```

3.9.11 CONFIG_PRL_SINKTX_TIMEOUT_MS

C

```
#define CONFIG_PRL_SINKTX_TIMEOUT_MS MILLISECONDS_TO_TICKS(16)
```

Description

CONFIG_PRL_SINKTX_TIMEOUT_MS defines the SinkTxTimer specified in the USB-PD Specification. Default value of CONFIG_PRL_SINKTX_TIMEOUT_MS is set as 16 milliseconds.

Remarks

CONFIG_PRL_SINKTX_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB PD Compliant.

Example

```
#define CONFIG_PRL_SINKTX_TIMEOUT_MS                    MILLISECONDS_TO_TICKS(16)
```

3.9.12 CONFIG_PRL_BIST_CONTMODE_TIMEOUT_MS

C

```
#define CONFIG_PRL_BIST_CONTMODE_TIMEOUT_MS MILLISECONDS_TO_TICKS(45)
```

Description

CONFIG_PRL_BIST_CONTMODE_TIMEOUT_MS defines the BISTContModeTimer specified in the USB-PD Specification. Default value of CONFIG_PRL_BIST_CONTMODE_TIMEOUT_MS is set as 45 milliseconds.

Remarks

CONFIG_PRL_BIST_CONTMODE_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB PD Compliant.

Example

```
#define CONFIG_PRL_BIST_CONTMODE_TIMEOUT_MS          MILLISECONDS_TO_TICKS(45)
```

3.9.13 CONFIG_PE_PSHARDRESET_TIMEOUT_MS

C

```
#define CONFIG_PE_PSHARDRESET_TIMEOUT_MS MILLISECONDS_TO_TICKS(28)
```

Description

CONFIG_PE_PSHARDRESET_TIMEOUT_MS defines the PSHardResetTimer specified in the USB-PD Specification. Default value of CONFIG_PE_PSHARDRESET_TIMEOUT_MS is set as 28 milliseconds.

Remarks

CONFIG_PE_PSHARDRESET_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB PD Compliant.

Example

```
#define CONFIG_PE_PSHARDRESET_TIMEOUT_MS              MILLISECONDS_TO_TICKS(28)
```

3.9.14 CONFIG_PE_PSTRANSITION_TIMEOUT_MS

C

```
#define CONFIG_PE_PSTRANSITION_TIMEOUT_MS MILLISECONDS_TO_TICKS(500)
```

Description

CONFIG_PE_PSTRANSITION_TIMEOUT_MS defines the PSTransitionTimer specified in the USB-PD Specification. Default value of CONFIG_PE_PSTRANSITION_TIMEOUT_MS is set as 500 milliseconds.

Remarks

CONFIG_PE_PSTRANSITION_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB PD Compliant.

Example

```
#define CONFIG_PE_PSTRANSITION_TIMEOUT_MS             MILLISECONDS_TO_TICKS(500)
```

3.9.15 CONFIG_PE_SENDERRESPONSE_TIMEOUT_MS

C

```
#define CONFIG_PE_SENDERRESPONSE_TIMEOUT_MS MILLISECONDS_TO_TICKS(26)
```

Description

CONFIG_PE_SENDERRESPONSE_TIMEOUT_MS defines the SenderResponseTimer specified in the USB-PD Specification. Default value of CONFIG_PE_SENDERRESPONSE_TIMEOUT_MS is set as 26 milliseconds.

Remarks

CONFIG_PE_SENDERRESPONSE_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB PD Compliant.

Example

```
#define CONFIG_PE_SENDERRESPONSE_TIMEOUT_MS           MILLISECONDS_TO_TICKS(26)
```

3.9.16 CONFIG_PE_SINKREQUEST_TIMEOUT_MS

C

```
#define CONFIG_PE_SINKREQUEST_TIMEOUT_MS MILLISECONDS_TO_TICKS(100)
```

Description

CONFIG_PE_SINKREQUEST_TIMEOUT_MS defines the SinkRequestTimer specified in the USB-PD Specification. Default value of CONFIG_PE_SINKREQUEST_TIMEOUT_MS is set as 100 milliseconds.

Remarks

CONFIG_PE_SINKREQUEST_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB PD Compliant.

Example

```
#define CONFIG_PE_SINKREQUEST_TIMEOUT_MS                MILLISECONDS_TO_TICKS(100)
```

3.9.17 CONFIG_PE_SINKWAITCAP_TIMEOUT_MS

C

```
#define CONFIG_PE_SINKWAITCAP_TIMEOUT_MS MILLISECONDS_TO_TICKS(465)
```

Description

CONFIG_PE_SINKWAITCAP_TIMEOUT_MS defines the SinkWaitCapTimer specified in the USB-PD Specification. Default value of CONFIG_PE_SINKWAITCAP_TIMEOUT_MS is set as 465 milliseconds.

Remarks

CONFIG_PE_SINKWAITCAP_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB PD Compliant.

Example

```
#define CONFIG_PE_SINKWAITCAP_TIMEOUT_MS                MILLISECONDS_TO_TICKS(465)
```

3.9.18 CONFIG_PE_SOURCECAPABILITY_TIMEOUT_MS

C

```
#define CONFIG_PE_SOURCECAPABILITY_TIMEOUT_MS MILLISECONDS_TO_TICKS(150)
```

Description

CONFIG_PE_SOURCECAPABILITY_TIMEOUT_MS defines the SourceCapabilityTimer specified in the USB-PD Specification. Default value of CONFIG_PE_SOURCECAPABILITY_TIMEOUT_MS is set as 200 milliseconds.

Remarks

CONFIG_PE_SOURCECAPABILITY_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB PD Compliant.

Example

```
#define CONFIG_PE_SOURCECAPABILITY_TIMEOUT_MS        MILLISECONDS_TO_TICKS(150)
```

3.9.19 CONFIG_PE_SRCRECOVER_TIMEOUT_MS

C

```
#define CONFIG_PE_SRCRECOVER_TIMEOUT_MS MILLISECONDS_TO_TICKS(800)
```

Description

CONFIG_PE_SRCRECOVER_TIMEOUT_MS defines the tSrcRecover specified in the USB-PD Specification. Default value of CONFIG_PE_SRCRECOVER_TIMEOUT_MS is set as 800 milliseconds.

Remarks

CONFIG_PE_SRCRECOVER_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB PD Compliant.

Example

```
#define CONFIG_PE_SRCRECOVER_TIMEOUT_MS                MILLISECONDS_TO_TICKS(800)
```

3.9.20 CONFIG_PE_VDMRESPONSE_TIMEOUT_MS

C

```
#define CONFIG_PE_VDMRESPONSE_TIMEOUT_MS MILLISECONDS_TO_TICKS(28)
```

Description

CONFIG_PE_VDMRESPONSE_TIMEOUT_MS defines the VDMResponseTimer specified in the USB-PD Specification. Default value of CONFIG_PE_VDMRESPONSE_TIMEOUT_MS is set as 28 milliseconds.

Remarks

CONFIG_PE_VDMRESPONSE_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB PD Compliant.

Example

```
#define CONFIG_PE_VDMRESPONSE_TIMEOUT_MS               MILLISECONDS_TO_TICKS(28)
```

3.9.21 CONFIG_PE_VCONNON_TIMEOUT_MS

C

```
#define CONFIG_PE_VCONNON_TIMEOUT_MS MILLISECONDS_TO_TICKS(100)
```

Description

CONFIG_PE_VCONNON_TIMEOUT_MS defines the tVCONNSourceOn specified in the USB PD Specification. Default value of CONFIG_PE_VCONNON_TIMEOUT_MS is set as 100 milliseconds.

Remarks

CONFIG_PE_VCONNON_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB PD.

Example

```
#define CONFIG_PE_VCONNON_TIMEOUT_MS                   MILLISECONDS_TO_TICKS(100)
```

3.9.22 CONFIG_PE_NORESPONSE_TIMEOUT_SECS

C

```
#define CONFIG_PE_NORESPONSE_TIMEOUT_SECS MILLISECONDS_TO_TICKS(5500)
```

Description

CONFIG_PE_NORESPONSE_TIMEOUT_SECS defines the NoResponseTimer specified in the USB-PD Specification. Default value of CONFIG_PE_NORESPONSE_TIMEOUT_SECS is set as 5.5 seconds.

Remarks

CONFIG_PE_NORESPONSE_TIMEOUT_SECS can be configured depending on the microcontroller platform for the device to be USB PD Compliant.

Example

```
#define CONFIG_PE_NORESPONSE_TIMEOUT_SECS        MILLISECONDS_TO_TICKS(5500)
```

3.9.23 CONFIG_PE_SRCTRANSISTION_TIMEOUT_MS

C

```
#define CONFIG_PE_SRCTRANSISTION_TIMEOUT_MS MILLISECONDS_TO_TICKS(28)
```

Description

CONFIG_PE_SRCTRANSISTION_TIMEOUT_MS defines the tSrcTransistionTimer specified in the USB-PD Specification.

Remarks

CONFIG_POWER_GOOD_TIMER_MS (see page 17) can be configured depending on the microcontroller platform for the device to be USB PD Compliant.

Example

```
#define CONFIG_PE_SRCTRANSISTION_TIMEOUT_MS              MILLISECONDS_TO_TICKS(30)
```

3.9.24 CONFIG_PE_VCONNOFF_TIMEOUT_MS

C

```
#define CONFIG_PE_VCONNOFF_TIMEOUT_MS MILLISECONDS_TO_TICKS(35)
```

Description

CONFIG_PE_VCONNOFF_TIMEOUT_MS defines the tVCONNOFF specified in the USB-Type C Specification. Default value of CONFIG_PE_VCONNOFF_TIMEOUT_MS is set as 35 milliseconds.

Remarks

CONFIG_PE_VCONNOFF_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB-Type C.

Example

```
#define CONFIG_PE_VCONNOFF_TIMEOUT_MS                MILLISECONDS_TO_TICKS(35)
```

3.9.25 CONFIG_PE_VCONNON_SELF_TIMEOUT_MS

C

```
#define CONFIG_PE_VCONNON_SELF_TIMEOUT_MS MILLISECONDS_TO_TICKS(150)
```

Description

CONFIG_PE_VCONNON_SELF_TIMEOUT_MS defines the tVCONNSourceOn specified in the USB PD Specification. Default value of CONFIG_PE_VCONNON_TIMEOUT_MS (see page 30) is set as 100 milliseconds.

Remarks

CONFIG_PE_VCONNON_SELF_TIMEOUT_MS can be configured depending on the microcontroller platform for the device to be USB PD.

Example

```
#define CONFIG_PE_VCONNON_SELF_TIMEOUT_MS                MILLISECONDS_TO_TICKS(150)
```